

PATENT ABSTRACTS OF JAPAN(11)Publication number : **2001-142937**(43)Date of publication of application : **25.05.2001**

(51)Int.Cl.

G06F 17/50(21)Application number : **2000-106543**(71)Applicant : **NEC CORP**(22)Date of filing : **07.04.2000**(72)Inventor : **ASHAR PRANAV
SUBURAJITTO BATACHARIYA
RAGHUNATHAN ANAND
GUPTA AARTI**

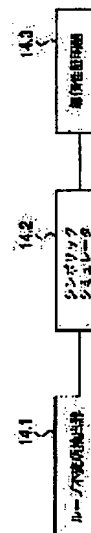
(30)Priority

Priority number : **1999 414815** Priority date : **08.10.1999** Priority country : **US****(54) SCHEDULING CORRECTNESS CHECKING METHOD AND SCHEDULE VERIFYING METHOD FOR CIRCUIT**

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a method for checking the correctness of scheduling of a circuit and a method for verifying the schedule of the circuit corresponding to the behavior description of the circuit.

SOLUTION: The schedule for the circuit is provided from the behavior description. Concerning the method for checking the correctness of scheduling of the circuit, a loop invariant term is extracted for determining the sufficient set of a non-cyclic thread while a loop is inside the circuit, a symbolic simulation is executed for extracting the loop invariant term, and the equivalency of the non-cyclic thread is proved. Concerning the method for verifying the schedule of the circuit corresponding to the behavior description, the schedule thread of possible execution containing the loop is selected from the schedule, a correspondent behavior thread is identified out of the behavior description, the un-conditional equivalency of the schedule thread and the behavior thread is proved, and the operation is repeated concerning all the threads of execution.

**LEGAL STATUS**

[Date of request for examination] 09.03.2001

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁 (J P) (12) 公開特許公報 (A)

(11)特許公開番号
特開2001-142937
(P2001-142937A)
(43)公開日 平成13年5月25日(2001.5.25)

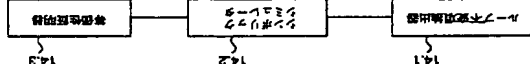
(51)Int.Cl.
G 0 6 F 17/50
F I
G 0 6 F 17/50
6 6 4
6 6 4 G 5 B 0 4 6
7-730-7(参考)

審査請求 未請求 請求項の数43 O L (全 37 頁)

(21)出願番号 特開2000-106543(P2000-106543)
(22)出願日 平成12年4月7日(2000.4.7)
(31)優先権主張番号 09/4114815
(32)優先日 平成11年10月8日(1999.10.8)
(33)優先権主張国 米国 (U S)
(71)出願人 00004237
日本電気株式会社
東京都港区芝五丁目7番1号
(72)発明者
ブラナブ・アシャー
アメリカ合衆国、ニュージャージー
08540 プリンストン、4 インディペン
デンス ウェイ、エヌ・イー・シー・ユ
ー・エス・イー・インク内
(74)代理人 100097157
弁理士 桂木 雄二

(54)【発明の名称】 回路のスケジューリング正当性チェック方法及びスケジューリング検証方法

(57)【要約】
【課題】 回路のスケジューリングの正当性をチェックする方法、及び、回路のビヘイビア記述に対して回路のスケジューリングを検証する方法を実現する。
【解決手段】 回路に対するスケジューリングはビヘイビア記述から得られる。回路のスケジューリングの正当性をチェックする方法は、ループが回路内にあるときに非巡回スレッドの十分なセットを決定するためにループ不要項を抽出し、ループ不要項を抽出するためにシンボリックシミュレーションを実行し、非巡回スレッドの等価性を証明する。回路のビヘイビア記述に対して回路のスケジューリングを検証する方法は、スケジューラからループを含む可能性のある実行のスケジューリングスレッドを選択し、ビヘイビア記述から対応するビヘイビアスレッドを識別し、スケジューリングスレッド及びビヘイビアスレッドの等価性等価性を証明し、実行のすべてのスレッドについて以上を繰り返す。



(2)

【特許請求の範囲】

【請求項1】 回路に対するスケジューリングがビヘイビア記述から得られる場合の当該回路のスケジューリングの正当性をチェックする方法において、

(a) ループが回路内にあるときに非巡回スレッドの十分なセットを決定するためにループ不要項を抽出するステップと、

(b) 前記ループ不要項を抽出するためにシンボリックシミュレーションを実行するステップと、

(c) 前記非巡回スレッドの等価性を証明するステップと、

からなることを特徴とする回路スケジューリング正当性チェック方法。

【請求項2】 前記ビヘイビア記述は、サイクル境界の導入によって変換されることを特徴とする請求項1記載の方法。

【請求項3】 前記ビヘイビア記述は、演算並べ替えによって変換されることを特徴とする請求項1記載の方法。

【請求項4】 前記ビヘイビア記述は、ループの展開、巻付け、折畳み及びバイグライン化によって変換されることを特徴とする請求項1記載の方法。

【請求項5】 前記ビヘイビア記述は、演算の投機実行によって変換されることを特徴とする請求項1記載の方法。

【請求項6】 前記ステップ(c)は、シンボリックシミュレーションを用いて実行されることを特徴とする請求項1記載の方法。

【請求項7】 回路のビヘイビア記述に対して回路のスケジューリングを検証する方法において、

(a) 前記スケジューラから、ループを含む可能性のある実行のスケジューリングスレッドを選択するステップと、

(b) 前記ビヘイビア記述から、対応するビヘイビアスレッドを識別するステップと、

(c) スケジューリングスレッド及びビヘイビアスレッドの無条件等価性を証明するステップと、

(d) 実行のすべてのスレッドについて前記ステップ(a)～(c)を繰り返すステップと、

からなることを特徴とする回路スケジューリング検証方法。

【請求項8】 前記スケジューラは、スケジューリング状態遷移グラフとして指定されることを特徴とする請求項7記載の方法。

【請求項9】 前記ビヘイビアは、ビヘイビア状態遷移グラフとして指定されることを特徴とする請求項7記載の方法。

【請求項10】 前記ステップ(c)は、

(i) 前記スケジューリングスレッドをスケジューリングラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、

(ii) 前記スケジューリング構造グラフと前記ビヘイビア

構造グラフの等価性をチェックするステップと、からなることを特徴とする請求項7記載の方法。

【請求項11】 回路のビヘイビア記述に対して回路のスケジューリングを検証する方法において、

(a) スケジューラをスケジューリング状態遷移グラフとして指定するステップと、

(b) 回路のビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、

(c) 前記スケジューリング状態遷移グラフから、実行のスケジューリングスレッドを選択するステップと、

(d) 前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを識別するステップと、

(e) 前記スケジューリングスレッドをスケジューリングラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、

(f) 前記スケジューリング構造グラフと前記ビヘイビア構造グラフの等価性をチェックするステップと、

(g) 実行のすべてのスレッドについて前記ステップ(c)～(f)を繰り返すステップと、

からなることを特徴とする回路スケジューリング検証方法。

【請求項12】 前記ステップ(i)は、

(i) 前記ビヘイビア状態遷移グラフ内の各ノードが該ノードの推移フェーズ内のすべてのノードの後にのみ現れるように、前記ビヘイビア構造グラフ内のすべてのノードを含む順序セットorder1を作成するステップと、

(ii) 前記ビヘイビア構造グラフ内の各ノードが該ノードの推移フェーズ内のすべてのノードの後にのみ現れるように、前記スケジューリング構造グラフ内のすべてのノードを含む順序セットorder2を作成するステップと、

(iii) order1をたどり、ビヘイビア構造グラフ内の基底変数を識別するステップと、

(iv) ビヘイビア構造グラフ内の非基底変数を基底変数で表すステップと、

(v) スケジューリング構造グラフ内の入力ノードに対する等価性リストを作成するステップと、

(vi) order2をたどり、order2内の各ノードを処理して、スケジューリング構造グラフの入力からスケジューリング構造グラフの出力へ等価性リストを伝搬させるステップと、

(vii) uをビヘイビア構造グラフ内の信号の識別子とし、cを等価性の条件を表す二分決定ダイアグラムであるとして、各等価性リスト内のエントリは対(u, c)であり、ビヘイビア構造グラフ内の対応する出力ノードで等価性が決定したかどうか、及び、対応する条件cがorder2内のプライマリ出力ノードに対するトートロジーであるかどうかをチェックするステップと、

(viii) order2内のすべての出力ノードについて、前記ステップ(vii)を繰り返すステップと、

(3)

(ix) すべての出力ノードが等価であることがわか
た場合に等価性を見つけるステップと、

からなることを特徴とする請求項11記載の方法。

【請求項13】 実行の巡回スレッドを有する可能性のある回路のスケジュールと該回路のビヘイビアとの間の

(a) スケジュールをスケジュール状態選択グラフとして表現するステップと

(b) ビヘイビアをビヘイピア状態遷移グラフとして表

(c) 前記スケジュール状態遷移グラフ内の強連結成分を識別するステップ

(d) 各強連結成分内の終了ノードを識別するステップ

(c) 前記スケジュール状態遷移グラフをつづいて、前記強連結成分を辿らないサブパスを併合するステップ

(1) 以前に選択されていないパスを選択するステップ

(g) 前記ステップ(f)で選択されたパスに対する得点に基づいて、図路を再組するステップと

(h) 選択されたパスを列挙するのに必要すべての状態遷移決定をカプセル化するパスシグナルを生成するための回路を構築。G-T 回路に追加するステップと、

(i) パスシグナルを用いて、制約されたシンボリックシミュレーションを実行してヒューピア状態遷移グラフ内の対応するパスを識別し、該パスに対する構造RTL

(j) 選択されたパスにおいて、以前に選択されていた強連結成分を選択するステップと、

(k) 選択されたバス内の選択された強連結成分に対する不変項を、対応セットのリストとして抽出するステップと。

(1) 対応セットのリストから1つの対応セットを選択するステップと、

(m) 選択された対応セットが、前のシンボリックシミュレーションの強連結成分カットにおいて得られる変数対応により小さい場合に、シンボリックシミュレーションを再実行するステップと、

(n) 対応セットのリスト内の各対応セットについて前記ステップ (i) ~ (m) を繰り返すステップと、

(○) 川口等価性条件が、バス条件以外の条件付きであるかどうかをテストするステップと、

(p) 前記ステップ (o) で前記出力等価性が条件付きである場合に非等価性を報告してこの方法を終了するステップと、

(q) 選択されたパス内のすべての強連結成分について
前記ステップ(j)～(p)を繰り返すステップと、

(r) 終了点が高々3度現れるようにルートからシンクへのすべてのパスについて前記ステップ (f) ~ (q)

(6)

て回路のスケジュールを検証するためのコンピュータシステム。
【請求項28】 回路のビヘイビア記述に対して回路のスケジュールを検証するための、プロセスサ及びメモリを有するコンピュータシステムにおいて、前記メモリは、前記コンピュータシステムが、

- (a) スケジュールをスケジュール状態遷移グラフとして指定するステップと、
- (b) 回路のビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、
- (c) 前記スケジュール状態遷移グラフから、実行のスケジュールスレッドを選択するステップと、
- (d) 前記ビヘイビア状態遷移グラフから、対応するビヘイビスレッドを識別するステップと、
- (e) 前記スケジュールスレッドをスケジュール構造グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、
- (f) 前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックするステップと、
- (g) 実行のすべてのスレッドについて前記ステップ(c) ~ (f) を繰り返すステップと

【請求項29】 前記命令は、前記コンピュータシステムが、

- (i) 前記ビヘイビア状態遷移グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記ビヘイビア構造グラフ内のすべてのノードを含む順序セット `arr1` を作成するステップと、
- (ii) 前記ビヘイビア構造グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記スケジュール構造グラフ内のすべてのノードを含む順序セット `arr2` を作成するステップと、
- (iii) `arr1` をたどり、ビヘイビア構造グラフ内の基底変数を識別するステップと、
- (iv) ビヘイビア構造グラフ内の非基底変数を基底変数で表すステップと、
- (v) スケジュール構造グラフ内の入力ノードに対する等価性リストを生成するステップと、
- (vi) `arr2` をたどり、`arr2` 内の各ノードを処理して、スケジュール構造グラフの入力からスケジュール構造グラフの出力へ等価性リストを伝搬させるステップと、
- (vii) 各等価性リスト内のエントリは対 (u, c) であり、uはビヘイビア構造グラフ内の番号の識別子であり、cは等価性の条件を表す二分決定ダイアグラムであり、cは等価性の条件を表す二分決定ダイアグラムであるとして、ビヘイビア構造グラフ内の対応する出力ノードで等価性が確認したかどうか、及び、対応する条件 `c` が `arr2` 内のブライヤリ出力ノードに対するトートロジーであるかどうかをチェックするステップと、
- (viii) `arr2` 内のすべての出力ノードについて前記ステップ (vii) を繰り返すステップと、
- (ix) すべての出力ノードが等価であることがわかった場合に等価性を記付したとするステップと、
- (x) 前記ステップ (i) を実行することを可能にする命令をさらに含むことを特徴とする請求項28に記載のコンピュータシステム。

【請求項30】 回路のスケジュールと該回路のビヘイビアとの間の等価性を検証するための、プロセスサ及びメモリを有するコンピュータシステムにおいて、前記スケジュール及び前記ビヘイビアは、実行の巡回スレッドを有する可能性があり、

- (a) スケジュールをスケジュール状態遷移グラフとして表現するステップと、
- (b) ビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、
- (c) 前記スケジュール状態遷移グラフ内の強連結成分を識別するステップと、
- (d) 各強連結成分内の終了ノードを識別するステップと、
- (e) 前記スケジュール状態遷移グラフをつづがして、前記強連結成分を通らないサブパスを併合するステップと、
- (f) 以前に選択されていないパスを選択するステップと、
- (g) 前記ステップ (f) で選択されたパスに対する構造RTL回路を取得するステップと、
- (h) 選択されたパスを列挙するのに必要なすべての状態遷移決定をカプセル化するパスシグナルを生成するための回路を構造RTL回路に追加するステップと、
- (i) パスシグナルを用いて、制約されたシンボリックシミュレーションを実行してビヘイビア状態遷移グラフ内の対応するパスを識別するステップと、
- (j) 選択されたパスにおいて、以前に選択されていない強連結成分を選択するステップと、
- (k) 選択されたパス内の選択された強連結成分に対する不変項を、対応セットのリストとして抽出するステップと、
- (l) 対応セットのリストから1つの対応セットを選択するステップと、
- (m) 選択された対応セットが、前のシンボリックシミュレーションの強連結成分カットにおいて得られる変数に対応より小さい場合に、シンボリックシミュレーションを再実行するステップと、
- (n) 対応セットのリスト内の各対応セットについて前記ステップ (l) ~ (m) を繰り返すステップと、

【請求項31】 前記命令は、前記コンピュータシステムが、

- (i) ビヘイビア状態遷移グラフの始状態を許容パスリストに割り当てるステップと、
- (ii) 許容パスリスト内で以前に動いていない状態を選択するステップと、
- (iii) ビヘイビア構造RTLを生成するステップと、
- (iv) 非解釈シンボリックシミュレーションを実行して、スケジュール構造RTL及びビヘイビア構造RTL内の対応する信号を識別するステップと、
- (v) 遷移条件とパスシグナルの論理値がゼロでない場合に、状態 `Sj` の新しいコピーを許容パスに追加するステップと、
- (vi) `Sj` から `Sj+1` の各出遷移ごとに前記ステップ (v) を繰り返すステップと、
- (vii) 許容パス内に残る動いていない状態のみが最終状態のインスタンズとなるまで、すべての訪れていない状態について前記ステップ (iii) ~ (vi) を繰り返すステップとを用いてステップ (i) を実行すること

を可能にする命令をさらに含むことを特徴とする請求項30に記載のコンピュータシステム。

【請求項32】 前記命令は、前記コンピュータシステムが、各ループごとに、

- (i) 各カットが前記ループの各実行の境界における変数値を表すような、スケジュール内のパスの構造RTL回路内の3個のカットを識別するステップと、
- (ii) ビヘイビアにおけるパスの構造RTL回路内の対応するカットを識別して、第1と第2のカットの間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックするステップと、
- (iii) スケジュール及びビヘイビアのRTL回路における対応するカットの各対における変数どうしの間の等価関係を識別するステップと、
- (iv) 最後のカットと最後の前のカットとの間の等価関係が同一であるかどうかをチェックするステップと、
- (v) 前記ステップ (iv) の関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットである場合、最後の前のカットにおける等価関係を複製し、1つ以上のループ実行について2つのRTL回路を識別して、前記ステップ (iii) から繰り返すステップと、
- (vi) 前記ステップ (iv) の関係が同一である場合、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットでない場合、最後の前のカットにおける等価関係を、等価関係セットの集合に追加し、1つ以上のループ実行について2つのRTL回路を展開して、前記ステップ (iii) から繰り返すステップと、
- (vii) 前記ステップ (iv) の関係が同一である場合、最後のカットにおける等価関係を、等価関係セットの集合に追加するステップと、
- (viii) 等価関係セットの最終集合を、不変項の所望の集合として指定するステップと、
- (ix) 等価関係セットの最終集合を、不変項の所望の集合として指定するステップと、
- (x) 前記ステップ (k) を実行することを可能にする命令をさらに含むことを特徴とする請求項30に記載のコンピュータシステム。

【請求項33】 コンピュータが回路のスケジューリングの正当性をチェックすることを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラムにおいて、

回路に対するスケジュールは、ビヘイビア記述から得られ、

- (i) 前記コンピュータコードは、
- (ii) ループが存在するときに非巡回スレッドの十分なセットを決定するためにループ不変項を抽出するコンピュータコードと、
- (iii) ループ不変項を抽出するためのシンボリックシミュレーションのコンピュータコードと、
- (iv) 非巡回スレッドの等価性を証明するコンピュータコードとを含むことを特徴とする、コンピュータが回路のスケジューリングの正当性をチェックすることを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品。

【請求項34】 前記ビヘイビア記述は、サイクル境界の挿入によって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

【請求項35】 前記ビヘイビア記述は、演算並べ替えによって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

【請求項36】 前記ビヘイビア記述は、ループの展開、巻付け、折込み及びパイプライン化によって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

(6)

(o) 出力等価性条件が、非等価性を報告するパス条件以外の条件付きであるかどうかをテストするステップと、

- (p) 前記ステップ (o) で前記出力等価性が条件付きである場合にこの検証を終了するステップと、
- (q) 選択されたパス内のすべての強連結成分について前記ステップ (i) ~ (n) を繰り返すステップと、
- (r) 終了点が高々3度現れるようにルートからシンクへのすべてのパスについて前記ステップ (i) ~ (q) を繰り返すステップとを用いて前記検証を実行すること

を可能にすることを特徴とする、回路のスケジュールと該回路のビヘイビアとの間の等価性を検証するためのコンピュータシステム。

【請求項31】 前記命令は、前記コンピュータシステムが、

- (i) ビヘイビア状態遷移グラフの始状態を許容パスリストに割り当てるステップと、
- (ii) 許容パスリスト内で以前に動いていない状態を選択するステップと、
- (iii) ビヘイビア構造RTLを生成するステップと、
- (iv) 非解釈シンボリックシミュレーションを実行して、スケジュール構造RTL及びビヘイビア構造RTL内の対応する信号を識別するステップと、
- (v) 遷移条件とパスシグナルの論理値がゼロでない場合に、状態 `Sj` の新しいコピーを許容パスに追加するステップと、
- (vi) `Sj` から `Sj+1` の各出遷移ごとに前記ステップ (v) を繰り返すステップと、
- (vii) 許容パス内に残る動いていない状態のみが最終状態のインスタンズとなるまで、すべての訪れていない状態について前記ステップ (iii) ~ (vi) を繰り返すステップとを用いてステップ (i) を実行すること

を可能にする命令をさらに含むことを特徴とする請求項30に記載のコンピュータシステム。

【請求項32】 前記命令は、前記コンピュータシステムが、各ループごとに、

- (i) 各カットが前記ループの各実行の境界における変数値を表すような、スケジュール内のパスの構造RTL回路内の3個のカットを識別するステップと、
- (ii) ビヘイビアにおけるパスの構造RTL回路内の対応するカットを識別して、第1と第2のカットの間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックするステップと、
- (iii) スケジュール及びビヘイビアのRTL回路における対応するカットの各対における変数どうしの間の等価関係を識別するステップと、
- (iv) 最後のカットと最後の前のカットとの間の等価関係が同一であるかどうかをチェックするステップと、
- (v) 前記ステップ (iv) の関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットである場合、最後の前のカットにおける等価関係を複製し、1つ以上のループ実行について2つのRTL回路を識別して、前記ステップ (iii) から繰り返すステップと、
- (vi) 前記ステップ (iv) の関係が同一である場合、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットでない場合、最後の前のカットにおける等価関係を、等価関係セットの集合に追加し、1つ以上のループ実行について2つのRTL回路を展開して、前記ステップ (iii) から繰り返すステップと、
- (vii) 前記ステップ (iv) の関係が同一である場合、最後のカットにおける等価関係を、等価関係セットの集合に追加するステップと、
- (viii) 等価関係セットの最終集合を、不変項の所望の集合として指定するステップと、
- (ix) 等価関係セットの最終集合を、不変項の所望の集合として指定するステップと、
- (x) 前記ステップ (k) を実行することを可能にする命令をさらに含むことを特徴とする請求項30に記載のコンピュータシステム。

【請求項33】 コンピュータが回路のスケジューリングの正当性をチェックすることを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラムにおいて、

回路に対するスケジュールは、ビヘイビア記述から得られ、

- (i) 前記コンピュータコードは、
- (ii) ループが存在するときに非巡回スレッドの十分なセットを決定するためにループ不変項を抽出するコンピュータコードと、
- (iii) ループ不変項を抽出するためのシンボリックシミュレーションのコンピュータコードと、
- (iv) 非巡回スレッドの等価性を証明するコンピュータコードとを含むことを特徴とする、コンピュータが回路のスケジューリングの正当性をチェックすることを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品。

【請求項34】 前記ビヘイビア記述は、サイクル境界の挿入によって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

【請求項35】 前記ビヘイビア記述は、演算並べ替えによって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

【請求項36】 前記ビヘイビア記述は、ループの展開、巻付け、折込み及びパイプライン化によって変換されることを特徴とする請求項33に記載のコンピュータプログラム製品。

(7)

11

【請求項37】 前記ビヘイビア記述は、消費の稼働実行によって変換されることを特徴とする請求項3記載のコンピュータプログラム製品。

【請求項38】 コンピュータが回路のビヘイビア記述に対して回路のスケジューラを検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品において、該コンピュータコードは、

前記コンピュータが、スケジューラをスケジューラ状態遷移グラフとして指定することを可能にするスケジューラ状態遷移グラフが、前記スケジューラ状態遷移グラフから、実行のスケジューラスレッドを選択することを可能にするスケジューラスレッドセレクタコードと、

前記コンピュータが、前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを選択することを可能にするビヘイビアスレッドセレクタコードと、前記コンピュータが、前記スケジューラスレッドをスケジューラ状態遷移グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換することを可能にするコンピュータコードと、

前記コンピュータが、前記スケジューラ状態遷移グラフと前記ビヘイビア構造グラフの等価性をチェックすることを可能にする等価性チェックコードと、

前記コンピュータが、回路のビヘイビア記述に対して回路のスケジューラを検証することと、前記コンピュータを含むコンピュータ可読媒体を有するコンピュータプログラム製品。

【請求項39】 コンピュータが回路のビヘイビア記述に対して回路のスケジューラを検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品において、

前記コンピュータコードは、前記コンピュータが、

(a) スケジューラをスケジューラ状態遷移グラフとして指定するステップと、

(b) 回路のビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、

(c) 前記スケジューラ状態遷移グラフから、実行のスケジューラスレッドを選択するステップと、

(d) 前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを識別するステップと、

(e) 前記スケジューラスレッドをスケジューラ状態遷移グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、

(f) 前記スケジューラ状態遷移グラフと前記ビヘイビア構造グラフの等価性をチェックするステップと、

(g) 実行のすべてのスレッドについて前記ステップ

12

(c) ~ (f) を繰り返すステップと、

を実行することを可能にすることを特徴とする、コンピュータが回路のビヘイビア記述に対して回路のスケジューラを検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品。

【請求項40】 前記コンピュータコードは、前記コンピュータが、

(i) 前記ビヘイビア状態遷移グラフ内の各ノードが該ノードの推移ファンクション内のすべてのノードの後にのみ現れるように、前記ビヘイビア構造グラフ内のすべてのノードを含む順序セット $arr1$ を作成するステップと、

(ii) 前記ビヘイビア構造グラフ内の各ノードが該ノードの推移ファンクション内のすべてのノードの後にのみ現れるように、前記スケジューラ状態遷移グラフ内のすべてのノードを含む順序セット $arr2$ を作成するステップと、

(iii) $arr1$ をたどり、ビヘイビア構造グラフ内の基底変数を識別するステップと、

(iv) ビヘイビア構造グラフ内の非基底変数を基底変数で表すステップと、

(v) スケジューラ構造グラフ内の入力ノードに対する等価性リストを構成するステップと、

(vi) $arr2$ をたどり、 $arr2$ 内の各ノードを処理して、スケジューラ構造グラフの入力からスケジューラ構造グラフの出力へ等価性リストを伝搬させるステップと、

(vii) 各等価性リスト内のイベントリは対 (u, c) であり、u はビヘイビア構造グラフ内の信号の識別子であり、c は等価性の条件を表す二分決定ダイアグラムであるとして、ビヘイビア構造グラフ内の対応する出力ノードで等価性が決定したかどうか、及び、対応する条件 c が $arr2$ 内のプライマリ出力ノードに対するポートロジであるかどうかをチェックするステップと、

(viii) $arr2$ 内のすべての出力ノードについて前記ステップ (vii) を繰り返すステップと、

(ix) すべての出力ノードが等価であることがわかった場合に等価性を見つけたとするステップと、

を用いて前記ステップ (i) を実行することを可能にすることを特徴とする請求項39記載のコンピュータプログラム製品。

【請求項41】 コンピュータが回路のスケジューラと該回路のビヘイビアとの間の等価性を検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品において、前記スケジューラ及び前記ビヘイビアは、実行の巡回スレッドを有する可能性があり、

前記コンピュータコードは、前記コンピュータが、

(a) スケジューラをスケジューラ状態遷移グラフとし

(8)

13

て表現するステップと、

(b) ビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、

(c) 前記スケジューラ状態遷移グラフ内の強連結成分を識別するステップと、

(d) 各強連結成分内の終了ノードを識別するステップと、

(e) 前記スケジューラ状態遷移グラフをつづいて、前記強連結成分を通らないサブパスを併合するステップと、

(f) (i) 以前に選択されていないバスを選択するステップと、

(g) 前記ステップ (i) で選択されたバスに対する構造RTL回路を算得するステップと、

(h) 選択されたバスを列挙するのに必要なすべての状態遷移決定をカプセル化するバスシグナルを生成するための回路を構造RTL回路に追加するステップと、

(i) バスシグナルを用いて、制約されたシンボリックシミュレーションを実行してビヘイビア状態遷移グラフ内の対応するバスを識別し、該バスに対する構造RTL回路を取得するステップと、

(j) 選択されたバスにおいて、以前に選択されていない強連結成分を選択するステップと、

(k) 選択されたバス内の選択された強連結成分に対する不変項を、対応セットのリストとして抽出するステップと、

(l) 対応セットのリストから1つの対応セットを選択するステップと、

(m) 選択された対応セットが、前のシンボリックシミュレーションの強連結成分において得られる変数対応より小さい場合に、シンボリックシミュレーションを再実行するステップと、

(n) 対応セットのリスト内の各対応セットについて前記ステップ (i) ~ (m) を繰り返すステップと、

(o) 出力等価性条件が、バス条件以外の条件付きであるかどうかをテストするステップと、

(p) 前記ステップ (o) で前記出力等価性が条件付きである場合に非等価性を報告してこの方法を終了するステップと、

(q) 選択されたバス内のすべての強連結成分について前記ステップ (j) ~ (p) を繰り返すステップと、

(r) 終了点が高々3度現れるようにループからシグナルのすべてのバスについて前記ステップ (i) ~ (q) を繰り返すステップと、

を実行することを可能にすることを特徴とする、コンピュータが回路のスケジューラと該回路のビヘイビアとの間の等価性を検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品。

【請求項42】 前記コンピュータコードは、前記コン

14

ピュータが、

(i) ビヘイビア状態遷移グラフの始状態を許容バーストに割り当てるステップと、

(ii) 許容バースト内で以前に動いていない状態を選択するステップと、

(iii) ビヘイビア構造RTLを生成するステップと、

(iv) 非線形シンボリックシミュレーションを共行して、スケジューラ構造RTL及びビヘイビア構造RTL内の対応する信号を識別するステップと、

(v) 遷移条件とバスシグナルの論理値がゼロでない場合に、状態 S_j の新しいコピーを許容バスに追加するステップと、

(vi) S_i から S_j への各出遷移ごとに前記ステップ (v) を繰り返すステップと、

(vii) 許容バスに残る動いていない状態のみが終状態のインスタンスとなるまで、すべての動いていない状態について前記ステップ (i) ~ (vi) を繰り返すステップと、

を用いて、前記ステップ (i) の制約されないシンボリックシミュレーションを実行することを可能にすることを特徴とする請求項41記載のコンピュータプログラム製品。

【請求項43】 前記コンピュータコードは、前記コンピュータが、各ループごとに、

(i) 各カットが前記ループの各行の境界における変数値を表すような、スケジューラ内のバスの構造RTL回路内の3個のカットを識別するステップと、

(ii) ビヘイビアにおけるバスの構造RTL回路内の対応するカットを識別して、第1と第2のカットの間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックするステップと、

(iii) スケジューラ及びビヘイビアのRTL回路における対応するカットの各対における変数どうしの間の等価関係を識別するステップと、

(iv) 最後のカットと最後の前のカットとの間の等価関係が同一であるかどうかをチェックするステップと、

(v) 前記ステップ (iv) の関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットである場合、最後の前のカットにおける等価関係を改良し、1つ以上のループ実行について2つのRTL回路を展開して、前記ステップ (i) ~ (v) から繰り返すステップと、

(vi) 前記ステップ (iv) の関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットでない場合、最後の前のカットにおける等価関係を、等価関係セットの集合に追加し、1つ以上のループ実行について2つのRTL回路を展開して、前記ステップ (i) ~ (v) から繰り返すステップと、

【請求項44】 前記コンピュータコードは、前記コン

(9)

15

(vii) 前記ステップ (iv) の関係が同一である場合、最後のカットにおける等価関係を、等価関係セットの集合に追加するステップと、
(viii) 等価関係セットの集合内で、他のエントリのスーパーセットであるすべてのエントリを削除するステップと、
(ix) 等価関係セットの最終集合を、不変項の所望の集合として指定するステップと、
を用いて前記ステップ (k) で不変項を抽出することを可能にすることを特徴とする請求項 4 に記載のコンピュータプログラム製品。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 [1. 発明の詳細な説明]
[1. 1 発明の属する技術分野] 本発明はハイレベル合成におけるスケジューリングステップの検証(verification)に関する。本発明の主要な焦点は、スケジューリングとともに実行される可能性が高いすべての代表的な変換を含むスケジューリング検証のための新技術にある。特に、本発明は、ループと、スケジューリング中に実行されるさまざまなループ変換を扱うことが可能な検証技術を提供する。

【0002】

【従来の技術】 [1. 2 従来の技術] 回路を抽出するまでの時間を短縮する手段として、ハイレベル化機からの合成が重要であることはよく認識されている。これにより、高速合成が可能になるに加えて、再使用の観点からのもうひとつの利点となる。Chrysalis (6)、Synopsis (8) 及びその他の多くの会社から提供されるツールを用いた組合せ論理の検証は、初期ネットリスト仕組に対して最終論理ネットリストの妥当性の検証を行う必要があるため、ハイレベル・ビヘイビア (動作) 記述から得られるレジスタトランスファレベル (RTL) のネットリストを検証するためのツールも必要となる。本発明は、検証を実行する技術を改善するためのものである。よく知られているように、シミュレーションは、正当性(correctness)を検証しないにもかかわらず時間がかかるために、検証ステラテジとして十分ではあり得ない。そこで、フ

ォーマル検証の方法論が必要となる。
【0003】 初期ビヘイビア仕組から最終RTLを実現するために適用される変換のスコープ (有効範囲) が与えられている場合、入力として単に2つの大欄に異なるレベルでの記述をとるブロックポック検証システムは、すべての実観的な目的で実現可能なわけではない。幸いに、合成自体は、自動ツールを用いてなされるか手動でなされるにかかわらず、一般的に、スケジューリング、リソース割当て及びレジスタ代入のような明確に区分された基本的なステップからなる共通の基本フローに依る。検証方法が実観的であるためには、このフロー

の知識を活用しなければならぬ。実観、スケジューリ

16

ングやレジスタ代入のようなステップどうしの間の区分をそのまま保持することは、「検証のための設計」の良イストラテジである。最終設計の品質が多少犠牲性になつたとしても、合成プロセスははるかに検証容易になる。
【0004】 ハイレベル合成フロー中の個々のステップの検証は、合成プロセス全体を検証するよりは容易であるものの、決して簡単ではない。スケジューリングと、タイムスタンプを演算(operation)に割り当てる作業である。同期設計では、これは、演算に状態を対応させることによって行われる。さまざまな設計要件を満たすために、演算並べ替え、ループ展開、技術実行 (speculative execution) 等のような変換が、このステップ中に実行されることがある。スケジューリングをチェクしようとする検証ツールにとって最小限の要件は、これらの変換をそのスコープ (有効範囲) に含むことである。

【0005】 本明細書において、シンボリックシミュレーションとは、回路を通して、変換ではなく、変換を伝搬させる手続きを含む。非解釈(uninterpret)ed) という用語は、この場合、標準の算術演算のような複雑な演算に遭遇したときに、入力のブール演算の値ではなく、入力リスト及び演算名が転送されることを意味する。

【0006】 1. 2. 1. 関連する研究
従来、ハイレベル記述から生成される設計を検証するためのいくつかの技術が提案されている。プログラム及びハードウェアの検証のためのシンボリックシミュレーションに関するかなりの研究活動が70年代及び80年代になされた。代表的なものとして、J. Darringer, "The application of program verification techniques to hardware verification", in Proc. Design Automation Conf., pp. 375-381, June 1979, を参照。しかし、Darringerの研究及びそこから派生した研究は、スケジューリングを検証するという場合に応用を限定している。派生した研究の一部は、

・R. Cory, "Symbolic simulation for functional verification with ADLiband SUL", in Proc. Design Automation Conf., pp. 82-89, June 1981
・V. Pitchumani and E. Stahler, "A formal method for computer design verification", in Proc. Design Automation Conf., pp. 809-814, June 1982

に見られる。
【0007】 重要な点として、Darringerの研究の主要な制限は、シンボリックシミュレータがチェクを実行するための不変項(invariant)を敷けることをユーザに要求していることである。実観、知られているように、2つのハードウェア記述を比較する際に、不変項は、一方の記述の完全な状態が他方の状態と一致しなければならぬ対応点 (Darringerの用語では制約点(control point)) である。スケジューリングの場合、シミュレータ

(10)

17

にこの情報を提供するために、ユーザは、例えば、合成ツールによって実行されるループ変換の詳細な知識を有する必要がある。このような要求は困難である。さらに、このような要求は、検証の目的に部分的に反することになる。また、ユーザが対応点を提供する場合、完全性の問題は未解決のままとなる。制約点としての間の関係番号の対応を検出し、それを利用して、制約点における同型(isomorphism)のためにチェクすべき式を単純化する追加能力を有する同じ基本的なアルゴリズムが、C.-T. Chen and A. Parker, "A hybrid numeric/symbolic program for checking functional and timing compatibility of synthesized design", in Proc. The International Symposium on High Level Synthesis, pp. 112-117, May 1994, で提案された。
【0008】 他のいくつかの関連する文獻についてもここで説明する。Minatoは、2つのハードウェア記述としての間の等価性を確かめるためのBDD (Binary Decision Diagram: 二分決定ダイアグラム) に基づくアプローチを提案している。S. Minato, "Generation of BDDs from hardware algorithm descriptions", in Proc. Int. Conf. Computer-Aided Design, pp. 644-649, Nov. 1996, を参照。このアプローチでは、すべての条件分岐は、追加変数の使用により直線的なコードに変換される。さらに、ループは、すべての変数に対するBDDが追加展開で変化しなくなるまで各ループを展開することによって処理される。この方法は、算術関数を表現する際のBDDの制限と、ループ終了条件が満たされるまでループを明示的に展開する必要があることによる欠点がある。Cong et al. は、ハイレベル合成におけるさまざまなステップをチェクするための規則スイートのセットを提案した。J. Cong, C. T. Chen, and K. Kucukcar, "Multi-dimensional rule checking for high-level design verification", in Proc. Int. High-level Design Validation & Test Wkshp., Nov. 1997, を参照。しかし、彼らの等価性チェッカーは、構造同型をチェクすることに制限されていた。Bergamaschi and Bajeの著論は、2つの記述における対応する信号が相異なる時点で報酬されなければならないときに示すことれば等価性チェクを実行することができると示している。R. A. Bergamaschi and S. Baje, "Observing the timeintervals: Verifying high-level synthesis results", IEEE Design & Test of Computers, vol. 8, pp. 40-50, Apr. 1997, を参照。

【0009】 最近では、検証において算術及び制御算術相互作用をモデル化するのためのいくつかの技術が提案されている。
・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

18

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

【0010】 この数年、非解釈関数によるシンボリックシミュレーションに基づく等価性チェクのための基本的アルゴリズムの効率の改善について、いくつかの論文が発表されている。
・R. Shostak, "An algorithm for reasoning about equality", Communications of the ACM, vol. 21, no. 7, pp. 583-585, 1978
・R. Jones, D. Dill, and J. Burch, "Efficient validity checking for processor validation", in Proc. Int. Conf. Computer-Aided Design, pp. 2-6, Nov. 1995
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

を参照。本発明で用いられるシンボリックシミュレーションアルゴリズムは、従来技術といくつかの共通点を有する。その決定手続きは、算術演算とともに、ブール演算を含む。
・C. Barrett, D. Dill, and J. Levitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996
・A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998

・K. T. Cheng and A. S. Krishnakumar, "Automatic functional test generation using the extended finite state machine model", in Proc. Design Automation Conf., June 1993

・F. Fallar, S. Deredas, and K. Keutzer, "Function al vector generation for IDL models using linear programming and 3-satisfiability", in Proc. Design Automation Conf., June 1998
・J. Kukula, T. Shiple, and A. Aziz, "Implicit state enumeration for FSDs with datapaths", in Proc. Formal Methods in Computer Aided Design, Nov. 1998

を参照。これらの技術は強力であり、ハイレベル合成から生成される設計の検証におけるモデルチェク技術や定理証明とともに、将来の応用の可能性がある。
・J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, "Symbolic model checking for sequential circuit verification", IEEE Transactions on Computer-Aided Design, vol. 13, Apr. 1994
・R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. Int. Conf. Computer-Aided Verification, July 1996
・S. Over, J. M. Rushby, and N. Shankar, "PVS: A prototype verification system", in 11th International Conference on Automated Deduction (D. Kapur, ed.), vol. 607 of Lecture Notes in Artificial Intelligence, Springer Verlag, 1992

(11)

19

を参照。また、要求に応じて、決定手続きに追加の代数を加えることも可能である。C. Barrett, D. Dill, and J. Lovitt, "Validity checking for combinations of theories with equality", in Proc. Formal Methods in Computer Aided Design, pp. 187-201, Nov. 1996, を参照。

[0011] しかしながら、本発明で用いられるシンボリックシミュレーションアルゴリズムは、ブール演算/条件をどのように扱うかにおいて従来技術とは異なる。最も近いのはA. Goel et al.のものであるが、対応する関数を記述するのに必要なブックキーベージにおいて異なる。A. Goel, K. Sajid, H. Thou, A. Aziz, and V. Singhal, "BDD based procedures for a theory of equality with uninterpreted functions", in Proc. Int. Conf. Computer-Aided Verification, pp. 244-255, July 1998, を参照。

[0012] 1. 2. 2 従来技術: スケジュールリングの有効範囲

スケジューリングは、ハイレベル合成に基づく設計フローにおいて最も重要なステップのうちの1つである。スケジューリングに関する全般的な情報については、

・D. Gajski, N. D. Dutt, A. C.-H. Wu, and S. Y. -L. Lin, High-level Synthesis: Introduction to Chip and System Design, Kluwer Academic Publishers, Norwell, MA, 1992

・G. De Micheli, Synthesis and Optimization of Digital Circuits, McGraw-Hill, New York, NY, 1994

を参照。タイミング情報を部分的にしか与えは含まないビヘイビア記述からはじめて、設計のサイクルコストのビヘイビアは、スケジューリングステップ中に固定される。このサブセクションでは、スケジューリングステップ中に実行されるいくつかの代表的な変換について説明する。それらの変換により検証プロセスの複雑さがどのように増大するかについてもここで説明する。

[0013] 1. 2. 2. 1 クロックサイクル境界の埋入: スケジューリングは、回路のビヘイビア記述からスケジューリングのプロセスである。単純な形のスケジューリングでは、実行されるのは、ビヘイビア記述にクロックサイクル境界、すなわちカットを入れることとかなる変換だけである。HDL記述の場合、これに相当する可能性のあるもの1つは、ビヘイビア記述にいくつかの"wait until clk=1 and clk_event"文を挿入することである。詳細は、D. Knapp, T. Ly, D. Muchillo, and R. Miller, "Behavioral synthesis methodology for HDL-based specification and validation", in Proc. Design Automation Conf., pp. 28-291, June 1995, を参照。あるサイクル境界と次のサイクル境界間の演算の列は組合せ論理を要するため、一般に、いくつかの条件を満たすために複数のカットを入れる。例えば、すべてのループ (VHDLのprocess文やVerilogの

20

alwaysブロックのような暗黙のループを含む) を引るためにカットを入れる。知られているように、ビヘイビアとスケジューリングとは、サイクルごとに等価ではない。従って、等価性の概念と、等価性をチェックする技術とは、クロックサイクル境界を超えて作用する必要がある。同様に、出力を計算するのに必要なクロックサイクル数は、異なるスレッドあるいは入力値に対しては異なる可能性がある。さらに、(データ依存性がある可能性もある) ループの存在もまた、検証の複雑さを増大させる。さらに、HDLの複雑なセマンティクス (例えば、信号代入や並行文) のため、サイクル境界を挿入するという単純な変換でさえ、設計の機能を変更することがある。これは、次の例によって明確に例示される。

[0014] 例1: 図1に示すVHDL記述を考える。この記述は、whileループと、さまざまな変数及び信号代入文を含むプロセスに関する。いくつかのステート until clk=1' and clk_event' イベント文に注釈を付けてある。これらのイベント文は、スケジューリング中に追加されたクロックサイクル境界を示す。なお、\$var, y \$var, \$var及び\$clkは信号であり、これらの変数に対してなされるすべての代入は信号代入である。VHDにおける信号代入文のセマンティクスは、信号に代入される値は即時に計算されるが、その代入はある後の時刻まで有効にならないというものである。この時刻は、明示的な時刻が指定されていない場合、デフォルトでは、デルタに等しい。"wait for 0ns;"文の目的は、デルタ遅延を導入し、先行する信号代入文によって生成された新しい値が有効になることを強制することである。[0015] ビヘイビア記述におけるwhileループ内の信号\$varへの代入を考える (なお、このビヘイビア記述は、"wait until clk=1' and clk_event"文を含んでいる)。右辺の式の計算は、信号\$varの古い値を使用する。信号\$varへの先行する代入は実行されているが、ループの最後の"wait for 0ns;"文まで有効ではないのである。しかし、スケジューリングでは、\$varへの信号代入の後に"wait until clk=1' and clk_event"文を導入することにより、\$varへの代入が評価される前に\$varの新しい値が有効になることが強制される。上記の並の結果として、スケジューリングは、シミュレーション中に置いた値を生産する可能性がある。

[0016] 1. 2. 2. 2 演算の並べ替え: 演算の並べ替えは、ビヘイビア記述に存在する並列性を利用するため、及び、与えられたリソースを最大限に利用するために、スケジューリング中に実行することが可能である。一般に、これは条件演算及び完全なループを並べ替えることを含む。最新のスケジューリング技術では、しばしば、データフロー及びメモリアクセスの依存性を維持しながら、ビヘイビア記述における演算を任意に並べ替える。詳細には、

21

・D. D. Gajski, N. D. Dutt, A. C.-H. Wu, and S. Y. -L. Lin, High-level Synthesis: Introduction to Chip and System Design, Kluwer Academic Publishers, Norwell, MA, 1992

・G. De Micheli, Synthesis and Optimization of Digital Circuits, McGraw-Hill, New York, NY, 1994

を参照。演算の並べ替え中に導入される可能性のあるエラーには、データ依存性、条件制御依存性、及びメモリアザード (例えば、RAW(read-after-write), WAW(write-after-write) など) の違反がある。演算の並べ替えにより生成されるようなスケジューリングの検証は、スケジューラからの制御及びデータのフローの抽出を必要とする。さらに、制御及びデータの依存性が実装において満たされることをチェックすること (例えば、構造同型チェック技術や初期チェック技術を用いて) が含まれる。詳細には、J. Gong, C. T. Chen, and K. Kulkarni, "Multi-dimensional rule checking for high-level design verification", in Proc. Int. High-level Design Validation & Test Workshop, Nov. 1997, を参照。

[0017] 例2: 図2 (a) はビヘイビアC記述を示し、図2 (b) はその対応するスケジューリングを示す。この例のビヘイビアは、シーケンシャルプログラムとして指定されているため、各スレッドで実行される演算について完全な順序を定義している。しかし、スケジューラは、保存の必要がある演算どうし間の依存性の解析を自動的に実行するかもしれない。演算の順序が出力の計算にとって重要でないときにも演算を並べ替えることを選択することがある。このような並べ替えは、リソースやクロック期間の数を最適にするために実行される可能性もある。

[0018] 以下の並べ替え操作が、このビヘイビアについてのスケジューリングで実行されている。

・ビヘイビアにおいて+2及び*1とマークされた演算の順序は逆転されている。これは、基本ブロック内の演算の局所並べ替えの例である。この並べ替えは正しくない。その理由は、ビヘイビアにおける演算+2と*1の間にデータ依存性があり (+2の出力は*1の入力である) 、このデータ依存性は、図2に示すスケジューリングでは破れているからである。

[0019] 2つのforループの実行順序はスケジューリングでは逆転されている。ビヘイビアにおいて最初に現れるループは、スケジューリングでは状態S2, S3, 及びS4によって実現され、ビヘイビア記述の第2のforループは、スケジューリングの状態S1で実現されている。この並べ替えは妥当である。その理由は、2つのループの間にデータ依存性や優先順位制約がないからである (これらのループに共通な唯一の変数であるループカウンタcountは、各ループの前に0に初期化される)。

[0020] 1. 2. 2. 3 パラメータメントの選

(12)

22

型: ビヘイビア記述における相異なるパス (すなわち、計算のスレッド) は、しばしば、異なるスケジューリングの機会及び制約を提示する。従って、ビヘイビアにおいて与えられたパスを最大限に最適化するためには、ビヘイビアにおける残りのパスとは別個にパス (またはその部分) をスケジューリングする必要があることはある。これにより、スケジューリングにおいてパスまたはセグメントの重複が生じる。パスに基づくスケジューリング技術は、ビヘイビアにおける単純 (無閉路) あるいは非巡回 (acyclic) パスに対してこのような最適化を行う。同様に、ループ指向スケジューリング技術は、ビヘイビアにおける非単純パスに対してこのような最適化を自動的に行う。

・R. Camposano, "Path-based scheduling for synthesis", IEEE Trans. Computer-Aided Design, vol. 10, pp. 85-93, Jan. 1991

・S. Bhattacharya, S. Iyer, and F. Brglez, "Performance analysis and optimization of schedules for conditional and loop-intensive specifications", in Proc. Design Automation Conf., pp. 491-496, June 1999

4

を参照。

[0021] また、スケジューリング中のパス/セグメントの複製は、検証プロセスの複雑さを増大させる。知られているように、演算と変数の間の関係は一対一ではなくなる。従って、構造同型をチェックする単純な技術は、スケジューリングとビヘイビアの等価性を証明するのに十分ではない。複製により、ビヘイビアに対するスケジューリングにおける演算の数が増大するが、ビヘイビア、あるいは、そのビヘイビア内の与えられたパスあるいはスレッドに沿って実行される演算のセットは同一である。このように、従来の検証ストラテジは、ビヘイビア及びスケジューリングにおけるパスを列挙することである。さらに、対応するパスのそれぞれの対ごとに、このようなストラテジは、ビヘイビア及びスケジューリングにおいて実行される演算のセットが同型のデータフローグラフを形成することを検証する。さらに詳細には、C.-T. Chen and A. Parker, "A hybrid numeric/symbolic program for checking functional and timing compatibility of synthesized designs", in Proc. The International Symposium on High-Level Synthesis, pp. 112-117, May 1994, を参照。

[0022] 1. 2. 2. 4 ループ変換: ループは、しばしば、ビヘイビア記述においてパフォーマンスのあるいはパワーに関するクリアルな部分を構成する。データ独立ループ (実行回数が事前に既知であり、入力値とは独立なループ) 、及び、データ依存ループ (実行回数が事前に既知ではなく、入力データに依存するループ) を積極的に最適化するさまざまなスケジューリング技術が提案されている。これらの技術には以下の

(13)

23

ものがある。

【0023】ループ展開、ループ展開の1つの意味は、ビヘイビアにおけるループをループ本体(loop body)のいくつかのコピーに変換した後、そのループのコピーをすることである。第2の意味は、スケジュールにおけるループの1回の実行が、ビヘイビアにおけるループの複数回の実行に対応することである。2種類のループ変換を図3(b)及び図3(c)に例示する。

【0024】ループ回転、これにより、スケジュールにおけるループの境界は、ビヘイビアにおける対応するループの境界に対してずれる。ループ回転を図3(d)に例示する。

【0025】ループノイズライニング、これは、ループ折込み(loop folding)あるいはループ巻付け(loop winding)ともいい、ループ本体の複数回の実行を並行して実行するものである。これには、正当性を保証するためにプロログ及びエピローグを作成することも必要になることがある。さらに詳細には、R. Potaman, J. Lis, A. Nicolau, and D. Gajski, "Termination based synthesis", in Proc. Design Automation Conf., pp. 444-449, June 1990, を参照。ループノイズライニングを図3(e)に例示する。

【0026】ビヘイビアにおけるループの存在と、スケジュール中のループ最適化の適用は、検証を非常に複雑にする。特に、ビヘイビア及びスケジュールにおけるスレッドあるいはバスの列挙は、ループの相異なる実行カウントを考慮する必要がある。さらに、ループが実行される回数はデータ依存であることがあり、静的に限定することが困難である。さらに、このような限定が可能である場合、あるいは、ループ実行回数が一定で既知である場合であっても、ビヘイビア及びスケジュールにおける異なるバスの個数により、すべてのこのようなバスの列挙は至難となる。さらに、回転やノイズライニングのようなループ最適化は、スケジュールとビヘイビアにおけるループの境界どうしの間の対応を破壊する。本発明の重要な特徴は、スケジュールにおけるすべての非単純バスの列挙を避ける、ループ不変項の自動抽出にある。

【0027】1. 2. 2. 5. 投機実行、投機実行では、ビヘイビア記述の一部が、その部分を実行することは必要であると思われる。投機実行は、ハイレベル合成のスケジュールングステップに統合されるとき、大幅なパフォーマンス改善が得られる。しかし、投機実行によって、検証は更に複雑になる。重要な点で、スケジュールでの制約依存性は、投機実行を含むスケジュールでは満たされない。さらに詳細には、

・I. Radivojevic and F. Brewer, "Ensemble representation and techniques for exact control-dependent scheduling", in Proc. High-level Synthesis Workshop, pp. 60-65, 1994

24

・O. Lakshminarayana, A. Raghunathan, and N. K. Jha, a, "Incorporating speculative execution into scheduling for control-flow intensive behaviors", in Proc. Design Automation Conf., pp. 108-113, June 1998 を参照。

【0028】スケジュラは、投機実行される演算の結果を格納するためにスケジュールに追加一時変数を導入するものが一般的である。また、スケジュラは、それら一時変数が依存する投機条件が評価された後にそれら一時変数を解放するための追加コード(代入文)を生成する。前述の問題に基づく検証技術は、このような変換を検証することができない。これについては、

・J. Gong, C. T. Chen, and K. Kucukcakar, "Multi-dimensional rule checking for high-level design verification", in Proc. Int. High-level Design Validation & Test Wkshp., Nov. 1997

・C.-T. Chen and A. Parker, "A hybrid numeric/symbolic program for checking functional and timing compatibility of synthesized design", in Proc. The 1st International Symposium on High Level Synthesis, pp. 112-117, May 1994 に説明されている。

【0029】

【発明が解決しようとする課題】[2. 発明の概要] 本発明は、新規な非線形シンボリックシミュレーション手続きに関する。本発明の技術は、ビヘイビア仕様及びスケジュールされたRTLが与えられた場合に、2つの記述の出力が相互に無条件に対応するかどうかを判定する。

【0030】スケジュールされたRTLとビヘイビア記述の間の、条件付きの可能性のある入力対応のリストからはじめて、本発明の技術は、2つの記述における信号の間の条件付き信号対応を出力へ向かって伝搬させる。2つの条件でその演算への入力が相互に対応する場合に、相互に対応する。その場合、出力が対応するため条件は、入力が対応するための条件の論理積となる。

【0031】算術演算とは異なり、ブール演算は完全に解釈される。これにより、条件を超えて演算を移動させるような変換の正当性のチェックが可能となる。このような変換は、スケジュールにおいて一般的である。【0032】スケジュールングを検証する作業は、ビヘイビア記述におけるループと、スケジュール中のループ変換とは異なる存在に、非常に複雑になる。本発明の重要な特徴は、ループと、スケジュールにおけるループ変換とを存在する場合に、等価性チェックによって、スケジュールとビヘイビアにおける信号の間の対応形式で、不変項の効率的な抽出を行うことである。本発明の技術は、ほとんどの設計における状態空間爆発は、制約状態よりもデータバスレジスタによって引き起こさ

(14)

25

れるという観測に、部分的に基づいている。スケジュールングにおける代数的な変換とみなされるものに基づいて、本発明の技術は、スケジュールングによって生成されるほとんどの設計を検証することが可能である。本発明の技術は、扱うことができないループ最適化に遭遇した場合に、照った否定(フォールスネガティブ)を報告するという点で、悲観的である。本発明の技術の詳細については、その応用の具体例とともに、セクション4で説明する。

【0033】シンボリックシミュレーションアルゴリズムは、本発明の重要な構成要素であるが、本発明の主要な貢献ではない。本発明の主要な貢献は、ループを扱うことが可能な、無階層グラフに対する基本的なシンボリックシミュレーションアルゴリズムの改善にある。

【0034】従来の方法における問題点を解決するため、本発明の目的は、スケジュールと、そのビヘイビア記述との等価性を証明する改善された方法を提供することである。本発明は、いかなるスケジュールにも制限されず、従来の技術の項で説明したいかなる最適化がなされたスケジュールでも使用可能である。なお、従来の技術の項で説明した最適化は単なる例示であり、本発明は、他の最適化技術を適用したスケジュールにも適用可能である。

【0035】ビヘイビアは、従来の任意の形式で指定することができる。これには、制御フローグラフ、データフローグラフあるいは制御/データフローグラフ(CD FG: control/data flow graph)及びビヘイビア(組)状態マシンが含まれるが、これらには限定されない。ビヘイビア合成についての詳細は、D. Knapp, T. L. y, D. MacMillan, and R. W. Miller, "Behavioral synthesis is methodology for HDL-based specification and validation", in Proc. Design Automation Conf., pp. 28-29, June 1995, を参照。

【0036】本発明は、ビヘイビア及びスケジュールにおけるブライマリア入力変数の間の対応が与えられていること、及び、出力変数の対応と、出力変数が同一の値を有すると期待される時刻が明確に指定されていることを仮定する。本発明は、複数のループ、ネストしたループ、及びデータ依存ループを含むビヘイビア及びスケジュールを処理する。

【0037】本発明の検証手続きの正確さ及び完全性を保証するために設計及び合成のプロログが満たすことが必要とされる仮定は以下の通りである。

【0038】ビヘイビア記述における演算は、スケジュールングプロセス中にアミミックエンティティとして扱われるもの(例えば、算術及び比較演算)と、分解または変換される可能性があるもの(例えば、ブール演算)とに分けることができる。例えば、ワードあるいはビットベクトル演算(例えば加算)は、スケジュールングプロセス中には、そのゲートレベル実装に分解されない

26

ことがある。アトミック演算と非アトミック演算に演算を分けることは任意性を伴うことがあるが、検証手続きに与えられることは必要である。この情報は、本発明の検証技術の主要な構成要素である非線形シンボリックシミュレーション手続により、どの演算を解釈しどの演算を非解釈のまま残すべきかを決定するために使用される。

【0039】スケジュールングプロセスは、アトミック演算の解釈から導き出される知識を使用しない。例えば、算術及び比較演算がアトミックであると言われる場合、スケジュールングは、スケジュール最適化するために、これらの演算の機能についての知識を使用しない。比較演算は、分岐及びループ終了条件を決定するために使用されるものを含む。

【0040】ビヘイビアにおける各ループごとに、スケジュールには少なくとも1つの対応するループがある。スケジュールにおけるループの1回の実行は、ビヘイビアにおけるループの1回以上の実行に対応する。この性質を満たさないスケジュールは、検証手続きによってエラーありとしてフラグが立てられる。なお、この仮定は、ループ本体あるいは境界がビヘイビアとスケジュールとで同一であることを要求するものではない。むしろ、これは、ループ境界がビヘイビアからスケジュールへと実行されているだけであり、その逆ではないことを意味する。

【0041】上記の仮定はそれほど制限的ではない。その理由は、これらの仮定は、リストスケジュール、強制的スケジュールング(force-directed scheduling)、バスに基づくスケジュールング、ループ指向スケジュールング(loop-directed scheduling)などのような周知のスケジュールングアルゴリズムを含む最も実際的なスケジュールング技術によって満たされるからである。これについては、

・D. D. Gajski, N. D. Dutt, A. C.-H. Wu, and S. Y. L. Lin, High-level Synthesis: Introduction to Chip and System Design, Kluwer Academic Publishers, Norwell, MA, 1992

・G. De Micheli, Synthesis and Optimization of Digital Circuits, McGraw-Hill, New York, NY, 1994 に示されている。

【0042】本明細書では、代表的なスケジュールング技術という用語は、上記の仮定を満たすスケジュールングのアルゴリズムあるいはツールを表すために使用する。

【0043】ループについて正当性をチェックする場合、本発明のアップロードはループ不変項を使用するものである。しかし、ループ停止の問題、すなわち、ループ本体の後のコードが実際に実行されるかどうかには特に対処しない。ある意味で、本発明では、すべての $n \geq 0$ について、 n 回の反復後の停止を考え、すべての場合に

(15)

27

等価性をチェックする。本発明のアプローチのこの特徴は強調しななければならない。すなわち、ループ未だのすべての反復回数に対する等価性がチェックされる。なお、算術演算を扱うために非解釈関数を使用するため、解釈された値（これがスケジューラによって利用されたかどうかにかかわらず）に依存する停止条件を考慮することはこのフレームワークでは不可能である。例えば、ループが6回実行される場合に限りエラが生じれば、終了条件のために、ループは2回より多くは決して実行されないとする。この場合、本発明の手続きは、フォーリスネガティブを報告することになる。その理由は、本発明は、2回の実行後の停止のみならず、 $n=6$ を含むすべての回数 n の後の停止を考慮するからである。ループ反復回数が一定の上限を有するような場合を早期停止(early termination)という。

【0044】
【問題を解決するための手段】本発明の目的を達成するため、回路のスケジューリングの正当性をチェックする方法が提供される。回路に対するスケジュールは、ビヘイビア記述から得られる。この方法は、ループが回路内にあるときに非巡回スレッドの十分なセツトを決定するためにループ不変項を抽出するステップと、ループ不変項を抽出するためにシンボリックシミュレーションを実行するステップと、非巡回スレッドの等価性を証明するステップとを有する。

【0045】好ましくは、ビヘイビア記述は、サイクル境界の導入によって変換される。

【0046】好ましくは、ビヘイビア記述は、演算並べ替えによって変換される。

【0047】好ましくは、ビヘイビア記述は、ループの展開、巻付け、折畳み及びバイライン化によって変換される。

【0048】好ましくは、ビヘイビア記述は、演算の段塊実行によって変換される。

【0049】本発明のもう1つの特徴によれば、回路のビヘイビア記述に対して回路のスケジュールを検証する方法が提供される。この方法は、前記スケジュールから、ループを含む可能性のある実行のスケジュールスレッドを選択するステップと、前記ビヘイビア記述から対応するビヘイビアスレッドを識別するステップと、スケジュールスレッドとビヘイビアスレッドの無条件等価性を証明するステップと、実行のすべてのスレッドについて繰り返すステップとを有する。

【0050】好ましくは、スケジュールは、スケジュール状態遷移グラフとして指定される。

【0051】好ましくは、ビヘイビアは、ビヘイビア状態遷移グラフとして指定される。

【0052】好ましくは、前記証明するステップは、前記スケジュールスレッドをスケジュール構造グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構

29

造グラフに変換するステップと、前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックするステップとを有する。

【0053】本発明のもう1つの特徴によれば、回路のビヘイビア記述に対して回路のスケジュールを検証する方法が提供される。この方法は、スケジュールをスケジュール状態遷移グラフとして指定するステップと、回路のビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、前記スケジュール状態遷移グラフから、実行のスケジュールスレッドを選択するステップと、前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを識別するステップと、前記スケジュールスレッドをスケジュール構造グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックするステップと、実行のすべてのスレッドについて繰り返すステップとを有する。

【0054】好ましくは、等価性チェックは、前記ビヘイビア状態遷移グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記ビヘイビア構造グラフ内のすべてのノードを含む順序セツト $a \ r \ i$ を作成するステップと、前記ビヘイビア構造グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記スケジュール構造グラフ内のすべてのノードを含む順序セツト $a \ e \ r \ i$ を作成するステップと、 $a \ e \ r \ i$ をたどり、ビヘイビア構造グラフ内の基底変数を識別するステップと、ビヘイビア構造グラフ内の非基底変数を基底変数で表すステップと、スケジュール構造グラフの入力ノードに対する等価性リストを構成するステップと、 $a \ r \ i$ をたどり、 $a \ e \ r \ i$ 内の各ノードを処理して、スケジュール構造グラフの入力からスケジュール構造グラフの出力へ等価性リストを伝搬させるステップと、各等価性リスト内のエントリは対 (u, c) であり、 u はビヘイビア構造グラフ内の信号の識別子であり、 c は等価性の条件を表す二分決定ダイアグラムであるとして、ビヘイビア構造グラフ内の対応する出力ノードで等価性が確定したかどうか、及び、対応する条件 c が $a \ r \ i$ 内のブライマリ出力ノードに対するトートルロジであるかどうかをチェックするステップと、 $a \ e \ r \ i$ のすべての出力ノードについて繰り返すステップと、すべての出力ノードが等価であることがわかった場合に等価性を認めたとするステップとを有するプロセスによって行われる。

【0055】本発明のもう1つの特徴によれば、回路のスケジュールと該回路のビヘイビアとの間の等価性を検証する方法が提供される。前記スケジュール及び前記ビヘイビアは、実行の巡回スレッドを有する可能性があり、前記方法は、スケジュールをスケジュール状態遷移グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、実行のすべてのスレッドについて繰り返すステップとを有する。

29

グラフとして表現するステップと、ビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、前記スケジュール状態遷移グラフ内の強連結成分を識別するステップと、各強連結成分内の終了ノードを識別するステップと、前記スケジュール状態遷移グラフをつぶして、前記強連結成分を辿らないサブパスを併合するステップと、以前に選択されていないパスを併合するステップと、選択されたパスに対する構造 $R \ T \ L$ 回路を取得するステップと、選択されたパスを列挙するのに必要なすべての状態遷移決定をカプセル化するパシシグナルを生成するための回路を構造 $R \ T \ L$ 回路に追加するステップと、パシシグナルを用いて、制約されたシンボリックシミュレーションを実行してビヘイビア状態遷移グラフ内の対応するパスを識別し、該パスに対する構造 $R \ T \ L$ 回路を取得するステップと、選択されたパスにおいて、以前に選択されていない強連結成分を選択するステップと、選択されたパス内の選択された強連結成分に対する不変項を、対応セツトのリストとして抽出するステップと、対応セツトのリストから1つの対応セツトを選択するステップと、選択された対応セツトが、前のシンボリックシミュレーションの強連結成分カッットにおいて得られる変数対応より小さい場合に、シンボリックシミュレーションを再実行するステップと、対応セツトのリスト内の各対応セツトについて以上のステップを繰り返すステップと、出力等価性条件が、パス条件以外の条件付きであるかどうかをテストするステップと、前記出力等価性条件付きである場合に非等価性を報告してこの方法を連結成分について以上のステップを繰り返すステップと、終了点が高々3度現れるようにルートからシンクへのすべてのパスについて以上のステップを繰り返すステップとを有する。

【0056】好ましくは、制約されないシンボリックシミュレーションが、ビヘイビア状態遷移グラフの始状態を許容パスリストに割り当てるステップと、許容パスリスト内で以前に訪れていない状態を選択するステップと、ビヘイビア構造 $R \ T \ L$ を生成するステップと、非解釈シンボリックシミュレーションを実行して、スケジュール構造 $R \ T \ L$ 及びビヘイビア構造 $R \ T \ L$ 内の対応する信号を識別するステップと、遷移条件とパシシグナルの編理情報でゼロでない場合に、状態 S_j の新しいコピーを許容パスに追加するステップと、 S_i から S_j への各出遷移ごとに前記追加するステップを繰り返すステップと、許容パス内に探索された訪れない状態のみが終状態のインスタンズとなるまで、すべての訪れない状態について繰り返すステップとを有するプロセスを用いて実行される。

【0057】好ましくは、不変項は、各ループごとに、各カッットが前記ループの各実行の境界における変数値を表すような、スケジュール内のパスの構造 $R \ T \ L$ 回路内

(16)

30

の3個のカッットを識別するステップと、ビヘイビアにおけるパスの構造 $R \ T \ L$ 回路内の対応するカッットを識別して、第1と第2のカッットの間のサブ回路と、第2と第3のカッットの間のサブ回路が同型であることをチェックするステップと、スケジュール及びビヘイビアの $R \ T \ L$ 回路における対応するカッットの各対における変数どうしの間の等価関係を識別するステップと、最後のカッットと最初の前のカッットとの間の等価関係が同一であるかどうかをチェックするステップと、前記関係が同一でなく、かつ、最後のカッットにおける等価関係が、最後の前のカッットにおける等価関係のサブセツトである場合、最後の前のカッットにおける等価関係を継承し、1つ以上のループ実行について2つの $R \ T \ L$ 回路を展開して、繰り送すステップと、前記関係が同一でなく、かつ、最後のカッットにおける等価関係が、最後の前のカッットにおける等価関係のサブセツトでない場合、最後の前のカッットにおける等価関係を、等価関係セツトの集合に追加し、1つ以上のループ実行について2つの $R \ T \ L$ 回路を展開して、繰り送すステップと、前記関係が同一である場合、最後の前のカッットにおける等価関係を、等価関係セツトの集合に追加するステップと、等価関係セツトの集合内で、他のエントリのスーパーセツトであるすべてのエントリを削除するステップと、等価関係セツトの最終集合を、不変項の希望の集合として指定するステップとを有するプロセスを用いて、ループから抽出される。

【0058】本発明のもう1つの特徴によれば、回路のスケジューリングの正当性をチェックするシステムが提供される。回路に対するスケジュールは、ビヘイビア記述から得られる。このシステムは、ループが存在するときに非巡回スレッドの十分なセツトを決定するループ不変項抽出器と、前記ループ不変項を抽出するシンボリックシミュレータと、非巡回スレッドの等価性を証明する等価性証明器とを有する。

【0059】好ましくは、前記ビヘイビア記述は、サイクル境界の導入によって変換される。

【0060】好ましくは、前記ビヘイビア記述は、演算並べ替えによって変換される。

【0061】好ましくは、前記ビヘイビア記述は、ループの展開、巻付け、折畳み及びバイライン化によって変換される。

【0062】好ましくは、前記ビヘイビア記述は、演算の段塊実行によって変換される。

【0063】本発明のもう1つの特徴によれば、回路のビヘイビア記述に対して回路のスケジュールを検証する方法が提供される。このシステムは、スケジュールをスケジュール状態遷移グラフとして指定するステップと、前記スケジュール状態遷移グラフ及びビヘイビア状態遷移グラフをつぶして、前記強連結成分を辿らないサブパスを併合するステップと、以前に選択されていないパスを併合するステップと、選択されたパスに対する構造 $R \ T \ L$ 回路を取得するステップと、選択されたパスを列挙するのに必要なすべての状態遷移決定をカプセル化するパシシグナルを生成するための回路を構造 $R \ T \ L$ 回路に追加するステップと、パシシグナルを用いて、制約されたシンボリックシミュレーションを実行してビヘイビア状態遷移グラフ内の対応するパスを識別し、該パスに対する構造 $R \ T \ L$ 回路を取得するステップと、選択されたパスにおいて、以前に選択されていない強連結成分を選択するステップと、選択されたパス内の選択された強連結成分に対する不変項を、対応セツトのリストとして抽出するステップと、対応セツトのリストから1つの対応セツトを選択するステップと、選択された対応セツトが、前のシンボリックシミュレーションの強連結成分カッットにおいて得られる変数対応より小さい場合に、シンボリックシミュレーションを再実行するステップと、対応セツトのリスト内の各対応セツトについて以上のステップを繰り返すステップと、出力等価性条件が、パス条件以外の条件付きであるかどうかをテストするステップと、前記出力等価性条件付きである場合に非等価性を報告してこの方法を連結成分について以上のステップを繰り返すステップと、終了点が高々3度現れるようにルートからシンクへのすべてのパスについて以上のステップを繰り返すステップとを有する。

【0064】好ましくは、制約されないシンボリックシミュレーションが、ビヘイビア状態遷移グラフの始状態を許容パスリストに割り当てるステップと、許容パスリスト内で以前に訪れていない状態を選択するステップと、ビヘイビア構造 $R \ T \ L$ を生成するステップと、非解釈シンボリックシミュレーションを実行して、スケジュール構造 $R \ T \ L$ 及びビヘイビア構造 $R \ T \ L$ 内の対応する信号を識別するステップと、遷移条件とパシシグナルの編理情報でゼロでない場合に、状態 S_j の新しいコピーを許容パスに追加するステップと、 S_i から S_j への各出遷移ごとに前記追加するステップを繰り返すステップと、許容パス内に探索された訪れない状態のみが終状態のインスタンズとなるまで、すべての訪れない状態について繰り返すステップとを有するプロセスを用いて実行される。

【0057】好ましくは、不変項は、各ループごとに、各カッットが前記ループの各実行の境界における変数値を表すような、スケジュール内のパスの構造 $R \ T \ L$ 回路内

(17)

31

るスケジュールスレッドをセレクトと、前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを選択するビヘイビアスレッドをセレクトと、前記スケジュールスレッドをスケジュール構造グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するコンパクタと、前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックする等価性チェックとを有する。

【0064】本発明のもう1つの特徴によれば、回路のスケジュールリングの正当性をチェックするための、プロセス及びメモリを有するコンピュータシステムが提供される。回路に対するスケジュールは、ビヘイビア記述から得られる。前記メモリは、前記コンピュータシステムが前記チェックを実行することを可能にする命令を含み、該命令は、ループが存在するときに非巡回スレッドの十分なセットを決定するためにループ不変項を抽出する命令と、ループ不変項を抽出するためのシンボリックシミュレーションの命令と、非巡回スレッドの等価性を証明する命令とを含む。

【0065】好ましくは、前記ビヘイビア記述は、サイクル境界の導入によって変換される。

【0066】好ましくは、前記ビヘイビア記述は、演算並べ替えによって変換される。

【0067】好ましくは、前記ビヘイビア記述は、ループの原則、巻付け、折畳み及びバイライン化によって変換される。

【0068】好ましくは、前記ビヘイビア記述は、演算の投機実行によって変換される。

【0069】本発明のもう1つの特徴によれば、回路のビヘイビア記述に対して回路のスケジュールを検証するための、プロセス及びメモリを有するコンピュータシステムが提供される。前記メモリは、前記コンピュータシステムが前記検証を実行することを可能にする命令を含み、該命令は、スケジュールをスケジュール状態遷移グラフとして指定する命令と、回路のビヘイビアをビヘイビア状態遷移グラフとして表現する命令と、前記スケジュール状態遷移グラフから、実行のスケジュールスレッドを選択する命令と、前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを選択する命令と、前記スケジュールスレッドをスケジュールをビヘイビア状態遷移グラフに変換する命令と、前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックする命令と、実行のすべてのスレッドについて繰り返す命令とを含む。

【0070】本発明のもう1つの特徴によれば、回路のビヘイビア記述に対して回路のスケジュールを検証するための、プロセス及びメモリを有するコンピュータシステムが提供される。前記メモリは、前記コンピュータシステムが、スケジュールをスケジュール状態遷移グラフ

32

フとして指定するステップと、回路のビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、前記スケジュール状態遷移グラフから、実行のスケジュールスレッドを選択するステップと、前記ビヘイビア状態遷移グラフから、対応するビヘイビアスレッドを識別するステップと、前記スケジュールスレッドをスケジュール構造グラフに変換するとともに前記ビヘイビアスレッドをビヘイビア構造グラフに変換するステップと、前記スケジュール構造グラフと前記ビヘイビア構造グラフの等価性をチェックするステップと、実行のすべてのスレッドについて繰り返すステップとを有することを可能にする命令を含む。

【0071】好ましくは、前記命令は、前記コンピュータシステムが、前記ビヘイビア状態遷移グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記ビヘイビア構造グラフ内のすべてのノード、前記ビヘイビア構造グラフ内の各ノードが該ノードの推移ファンイン内のすべてのノードの後にのみ現れるように、前記スケジュール構造グラフ内のすべてのノードを含む順序セット $arr2$ を作成するステップと、 $arr1$ をたどり、ビヘイビア構造グラフ内の基底変数を識別するステップと、ビヘイビア構造グラフ内の基底変数を基底変数で置き換えたかどうか、及び、対応する非基底変数の入力ノードに対する等価性リストを生成するステップと、 $arr2$ をたどり、 $arr2$ 内の各ノードを処理して、スケジュール構造グラフの入力からスケジュール構造グラフの出力へ等価性リストを伝搬させるステップと、各等価性リスト内のエントリは対 (u, c) であり、 u はビヘイビア構造グラフ内の信号の識別子であり、 c は等価性の条件を表す二分決定ダイアグラムであるとして、ビヘイビア構造グラフ内の対応する出力ノードで等価性が検証したかどうか、及び、対応する条件 c が $arr2$ 内のブライマリー出力ノードに対するポートローであるかどうかをチェックするステップと、 $arr2$ 内のすべての出力ノードについて繰り返すステップと、すべての出力ノードが等価であることがわかった場合に等価性を見つけたとするとステップとを実行することを可能にする命令とをさらに含む。

【0072】本発明のもう1つの特徴によれば、回路のスケジュールと該回路のビヘイビアとの間の等価性を検証するための、プロセス及びメモリを有するコンピュータシステムが提供される。前記スケジュール及び前記ビヘイビアは、実行の巡回スレッドを有する可能性がある。前記メモリは、前記コンピュータシステムが、スケジュールをスケジュール状態遷移グラフとして表現するステップと、ビヘイビアをビヘイビア状態遷移グラフとして表現するステップと、前記スケジュール状態遷移グラフ内の基底変数を基底変数で置き換えたかどうか、及び、対応する非基底変数の入力ノードに対する等価性リストを生成するステップと、 $arr2$ をたどり、 $arr2$ 内の各ノードを処理して、スケジュール構造グラフの入力からスケジュール構造グラフの出力へ等価性リストを伝搬させるステップと、各等価性リスト内のエントリは対 (u, c) であり、 u はビヘイビア構造グラフ内の信号の識別子であり、 c は等価性の条件を表す二分決定ダイアグラムであるとして、ビヘイビア構造グラフ内の対応する出力ノードで等価性が検証したかどうか、及び、対応する条件 c が $arr2$ 内のブライマリー出力ノードに対するポートローであるかどうかをチェックするステップと、 $arr2$ 内のすべての出力ノードについて繰り返すステップと、すべての出力ノードが等価であることがわかった場合に等価性を見つけたとするとステップとを実行することを可能にする命令とをさらに含む。

【0073】好ましくは、前記命令は、前記コンピュータシステムが、各ループごとに、各カットが前記ループの各実行の境界を越える変数を表示するような、スケジュール内のパスの構造 R 、 T 、 L の各のカットを識別するステップと、ビヘイビアにおける3個の構造 R 、 T 、 L の間の対応するカットを識別して、第1と第2のカットの間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックすることを可能にする命令とをさらに含む。

33

ール状態遷移グラフをつづき、前記強連結成分を通してないサブパスを併合するステップと、以前に選択されていないパスを選択するステップと、選択されたパスに対する構造 R 、 T 、 L の回路を取得するステップと、選択されたパスを列挙するのに必要なすべての状態遷移条件をカプセル化するパスシグナルを生成するための回路を構造 R 、 T 、 L の回路に追加するステップと、パスシグナルを用いて、制約されたシンボリックシミュレーションを実行してビヘイビア状態遷移グラフ内の対応するパスを識別するステップと、選択されたパスにおいて、以前に選択されてない強連結成分を選択するステップと、選択されたパス内の選択された強連結成分に対する不変項を、対応セットのリストとして抽出するステップと、対応セットのリストから1つの対応セットを選択するステップと、選択された対応セットが、前のシンボリックシミュレーションの強連結成分カットにおいて得られる変数対応より小さい場合に、シンボリックシミュレーションを再実行するステップと、対応セットのリスト内の各対応セットについて以上のステップを繰り返すステップと、出力等価性条件が、非等価性を報告するパス条件以外の条件付きであるかどうかをテストし、前記出力等価性が条件付きである場合にはこの検証を終了するステップと、選択されたパス内のすべての強連結成分について以上のステップを繰り返すステップと、終了点が高々3度現れるようにループからシンクへのすべてのパスについて以上のステップを繰り返すステップと、前記強連結成分を用いて前記検証を実行することを可能にする。

【0073】好ましくは、前記命令は、前記コンピュータシステムが、ビヘイビア状態遷移グラフの始状態を許容パスリストに割り当てるステップと、許容パスリスト内で以前に訪れない状態を選択するステップと、ビヘイビア構造 R 、 T 、 L を生成するステップと、非解読シンボリックシミュレーションを実行して、スケジュール構造 R 、 T 、 L 及びビヘイビア構造 R 、 T 、 L 内の対応する信号を識別するステップと、遷移条件とパスシグナルの論理値がゼロでない場合に、状態 S_i の新しいコピーを許容パスに追加するステップと、 S_i から S_j への各出遷移ごとに前記追加するステップを繰り返すステップと、許容パス内に残る訪れない状態のみが終状態のインスタンズとなるまで、すべての訪れない状態について繰り返すステップと、実行することを可能にする命令とをさらに含む。

【0074】好ましくは、前記命令は、前記コンピュータシステムが、各ループごとに、各カットが前記ループの各実行の境界を越える変数を表示するような、スケジュール内のパスの構造 R 、 T 、 L の各のカットを識別するステップと、ビヘイビアにおける3個の構造 R 、 T 、 L の間の対応するカットを識別して、第1と第2のカットの間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックすることを可能にする命令とをさらに含む。

(18)

34

ュール及びビヘイビアの R 、 T 、 L の間における対応するカットの各対における変数 d 以上の間の等価関係を識別するステップと、最後のカットと最後の前のカットとの間の等価関係が同一であるかどうかをチェックするステップと、前記関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係と異なる場合に、最後の前のカットにおける等価関係を、 1 以上のループ実行について 2 つの R 、 T 、 L の回路を展開して、繰り返すステップと、前記関係が同一でなく、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットでない場合に、最後の前のカットにおける等価関係を、等価関係の集合に追加し、 1 つ以上のループ実行について 2 つの R 、 T 、 L の回路を展開して、繰り返すステップと、前記関係が同一である場合、最後のカットにおける等価関係を、等価関係の集合に追加し、他のエントリのサブセットであるすべてのエントリを削除するステップと、等価関係の集合の最終集合を、不変項の集合として指定するステップとを実行することを可能にする命令とをさらに含む。

【0075】本発明のもう1つの特徴によれば、コンピュータが回路のスケジュールの正当性をチェックすることを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品が提供される。回路に対するスケジュールは、ビヘイビア記述から得られる。前記コンピュータコードは、ループが存在するときに非巡回スレッドの十分なセットを決定するためにループ不変項を抽出するコンピュータコードと、ループ不変項を抽出するためのシンボリックシミュレーションのコンピュータコードと、非巡回スレッドの等価性を証明するコンピュータコードとを含む。

【0076】好ましくは、前記ビヘイビア記述は、サイクル境界の導入によって変換される。

【0077】好ましくは、前記ビヘイビア記述は、演算並べ替えによって変換される。

【0078】好ましくは、前記ビヘイビア記述は、ループの原則、巻付け、折畳み及びバイライン化によって変換される。

【0079】好ましくは、前記ビヘイビア記述は、演算の投機実行によって変換される。

【0080】本発明のもう1つの特徴によれば、コンピュータが回路のビヘイビア記述に対して回路のスケジュールを検証することを可能にするコンピュータコードを含むコンピュータ可読媒体を有するコンピュータプログラム製品が提供される。前記コンピュータコードは、前記コンピュータが、スケジュールをスケジュール状態遷移グラフとして指定することを可能にするスケジュール状態遷移グラフジェネレータコードと、前記コンピュータが、回路のビヘイビアをビヘイビア状態遷移グラフと

(19)

35

[illegible]

【0081】本発明のむしろの特徴によれば、コンピュータが回路のビヘイビヤ迅速に對して回路のスケジューリングを検証することを可能にするコンピュータコードプログラム製品を提供される。前記コンピュータコードは、前記コンピュータ製品が提供される。前記コンピュータコードは、前記コンピュータが、スケジューリングのスケジューリング状態遷移グラフとして指定するステップと、回路のビヘイビヤをビヘイビヤ状態遷移グラフとして表現するステップと、前記スケジューリング状態遷移グラフから、実行のスケジューリングスレッドを選択するステップと、前記ビヘイビヤ状態遷移グラフから、対応するビヘイビヤスレッドを実行するステップと、前記スケジューリングスレッドをスケジューリンググラフに変換するとともに前記ビヘイビヤスレッドをビヘイビヤ構造グラフに渡換するステップと、前記スケジューリング構造グラフと前記ビヘイビヤ構造グラフの等価性をチェックするステップと、実行のすべてのスレッドについて繰り返すステップと、実行することとを可能にする。

[illegible]

(20)

37

前記コンピュータが、ベイズ状態遷移グラフの始状態を許容パスリストに割り当てするステップと、許容パスリストと、ベイズ以前に訪れない状態を選択するステップと、ベイズ以前に訪れない状態を選択するステップと、非解収シミュレーションを実行して、スケジューリング構造 $R_T L$ 及びベイズ構造 $R_T L$ 内の対応する相手を識別するステップと、遷移条件とパスシミュナルの論理積がゼロでない場合に、状態 S_j の新しいコピーを許容パスに追加するステップと、 S_j から S_j の各遷移ごとに前記追加するステップを繰り返すステップと、許容パス内に残る訪れない状態のみが終状態のインテグレーションとなるまで、すべての訪れない状態について繰り返すステップとを用いて、制約されないシミュレーションを実行することを可能にする。

【0085】好ましくは、前記コンピュータコードは、前記コンピュータが、各グループごとに、各カットが前記グループの各実体の境界における変数値を表すような、スケジュールの内パスの情報とR T L回路内の3個のカットの間の対応するステップと、およびピアにおけるパスの情報を識別するステップとを認識して、第1と第2 R T L回路内の対応するカットを識別して、第1と第2 R T L回路の間のサブ回路と、第2と第3のカットの間のサブ回路が同型であることをチェックするステップと、スケジューリング及びピーベンチのR T L回路における対応するカットの各対における変数値どうしの間の等価関係を識別するステップと、最後のカットと最後の前のカットの間の等価関係が同一であるかどうかをチェックすることと、前記関係が異なるかどうか、かつ、最後のカットにおける等価関係が、最後の前のカットにおける等価関係のサブセットである場合、最後の前のカットにおける

等価関係を検査し、1つ以上のループ実行について2つのR/T巡回を照会し、繰り返すステップと、前記関係が同一でなく、かつ、最後の前のカットにおける等価関係のサブセットである場合、最後の前のカットにおける等価関係を、等でない場合、最後の前のカットにおける等価関係を、等価関係セットの集合に追加し、1つ以上のループ実行について2つのR/T巡回を照会し、繰り返すステップと、前記関係が同一である場合、最後の前のカットにおける等価関係を、等価関係セットの集合に追加するステップと、等価関係セットの集合内で、他のエントリのサブセットであるすべてのエントリを削除するステップと、等価関係セットの最終集合を、不要の希望の集合として指定するステップを用いて不変項を抽出することを可能にする。

【0086】
【発明の実施の形態】 [4. 好ましい実施形態]
[4. 1 非巡回の場合の非解算問題によるシンボリックシミュレーション] (4. 1) では、有限長の非巡回インレットを有するスケジューラの場
合に、ビヘイビア記述に対してスケジューラを検証する際
の、非解算シンボリックシミュレーションアルゴリズム

の識別子であり、 c は等価性の条件を装する二分決定ダイアグラムであるとして、ビナビア構造グラフ内の対応する出力ノードが等価性が確定したかどうか、及び、対応する条件 c が a, r 2内の出力ノード出力ノードに属するトローゼーであるかどうかをチェックするステップと、 a, r 2内のすべての出力ノードについて繰り返すステップと、すべての出力ノードが等価であることがわかった場合に等価性を見つけたとするステップとを実行することを可能にする。

[illegible]

実行してビヘビヤ状態遷移グラフ内の対応するパスを識別し、該パスに対する構造R L回路を取得するステップと、選択されたパスにおいて、以前に選択されていた強連結成分を選択するステップと、選択されたパス内での選択された強連結成分に対する不変項を、対応セットのリストと対応するステップと、対応セットのリストから1つの対応セットを選択するステップと、選択された対応セットが、前のシンボリックジョーシンの強連結成分カットにおいて得られている変数に対応より小さい対応セットのリストにおいて得られている変数に対応するにたい場合、シンボリックジョーシンの再実行について、ステップと、対応セットのリスト内の各対応セットと、出力等価性条件が、パス条件以外の条件付きであるかどうかをテストするステップと、前記出力等価性が条件付きである場合と、選択されたパス内のすべての強連結成分について以上上のステップを繰り返すステップと、終了点が適々3度現れるようにループからシンクへのすべてのパスについて、以上のステップを繰り返すステップとを実行すること

を可能にする。

[0084] 好ましくは、前記コンピュータコードは、

(21)

40

は、構造グラフに変換される。

【0097】定数1 (構造グラフ)：構造グラフとは、有向グラフ $G=(V, A)$ であって、頂点のセット V は演算を実行するハードウェアコンポーネントを表し、辺のセットはコンポーネントの構造連結性を表すものである。頂点 $\in V$ は、型属性を有し、これは以下の値をとる。

- ・IN (ブライマリ入力変数と、レジスタ変数の現サイクル値を表す)
- ・OUT (ブライマリ出力変数と、レジスタ変数の現サイクル値を表す)
- ・OP (算術演算及び比較演算を含む、アトミックなワードレベル演算を表す)
- ・LOGIC (制御またはランダム論理を表す)
- ・MUX

構造グラフ内の辺にはそのビット幅が標記(annotate)される。

【0098】計算のセットから構造グラフを構成するプロセスは、ハードウェア記述言語 (HDL: hardware description language) からハードウェア構造を推論することと類似している。IN及びOUTノードは、ブライマリ入力変数、定数値、及び、レジスタ変数の現サイクル値及びサイクルの値を表すように生成される。OPノードは、ワードレベル計算及び条件演算 (例えば、比較演算、case演算など) に関連する代入演算に対応して生成される。単一ビットまたはビットベクトルに対するワード演算の使用により、構造グラフ内のLOGICノードが生成される。MUXノードは、相異なる代入演算が、相異なる条件下で同じ変数に代入を行うときに構成される。これらの条件に対応するOPまたはLOGICノードの出力は、与えられたクロックサイクルにおいて実行される代入を決定するために、MUXノードへの選択 (セレクト) 入力として使用される。

【0099】STG内の実行のスレッドと、等価であることを証明することが要求されるBSTG内の対応する実行のスレッド T' とが与えられると、各スレッドに沿って実行される計算はまず構造グラフに変換される。こうして、問題は、2つの構造グラフ $SSGT$ と $BSGT$ の等価性を証明することに帰着する。

【0100】このセクションの残りの部分では、以下の性質を利用した、構造グラフの等価性チェックのためのアルゴリズムの好ましい実施例について説明する。

・ビヘイビア記述からスケジュールを生成するときにOPノードのアトミック性は保存される。

・算術変換 (例えば、分配則や、乗算をシフトと加算で置き換えることなど) は実行されないということ。

【0101】定数2 (条件付き等価性)： $SSGT$ 内の信号 v が $BSGT$ 内の信号 u 、 u_2, \dots, u_n に条件付き等価であるとは、対応する条件 c_1, c_2, \dots, c_n (条件とは、 $BSGT$ あるいは $SSGT$ 内の入力変数へ

41

の代入の空でないセットを表す) であって、条件 c_k の下で、 $SSGT$ 内の信号 v における値が、 $BSGT$ 内の信号 u_k における値と等しいことが保証されるような条件 c_1, c_2, \dots, c_n が存在する場合をいう。条件付き等価関係を表すために、記法

【数1】

$$V \cong \{u_1, c_1, \dots, u_n, c_n\}$$

を用いる。

【0102】BDDは、条件付き等価関係に関連する条件を表すために使用される。一般に、条件自体は、入力変数で表すことも可能であり、また、さまざまな算術及び条件演算の結果を含むことも可能である。しかし、条件は、INノードに加えて、OP及びMUXノードの出力で (これらをまとめて、基底変数で) という表現される。実際、BDDは、閉閉論理に対してのみ構成される。これは、POに送られる次状態論理 R_{state_next} と、MUXノードを通るどのパスがセンシティブ化されているかあるいはマルチファンクションFUFがどのような設定されているかを決定する論理とを含む。

【0103】 $BSGT$ と $SSGT$ を比較するアルゴリズムの好ましい実施例の疑似コードを図4に示す。アルゴリズムは、 $SSGT$ と $BSGT$ のINノードどうしの間の等価関係から始める。このアルゴリズムは、POノードに到達するまで $SSGT$ 内の中間信号を通じて条件付き等価関係を生成し伝搬させ、 $SSGT$ 及び $BSGT$ における出力信号どうしの間の無条件等価性をチェックする。

【0104】まず、順序セット $Arr1$ ($Arr2$) を、 $BSGT$ ($SSGT$) 内のすべてのノードを含むように構成する。後方深さ優先探索走査を用いて、各ノードは、その推移ファンイン (transitive fanin) 内のすべてのノードの後にのみ現れるようにされる。次に、 $BSGT$ 内の基底変数 c 、PI、OP、及びMUXノードの出力として識別する。次に、 $Arr1$ を通る走査を実行し、基底変数に対応しない出力を有する各ノード (すなわち、各LOGICノード) について、そのノードの出力に対するBDD S を、その入力におけるBDDに隣りて取得する。各 $SSGT$ ノードは、その出力と、 $BSGT$ 内の信号との間の条件付き等価関係を表す等価性リストに関連づけられる。等価性リスト内のエントリは対 (u, c) である。ただし、 u は $BSGT$ 信号の識別子であり、 c は、等価性のための条件を表すBDDである。 $BSGT$ と $SSGT$ のINノードどうしの間の対応を用いて、 $SSGT$ のINノードに対する等価性リストを生成する。次に、 $Arr2$ を走り、各ノードを、その入力から出力へ等価性リストを伝搬させるように処理する。OP、LOGIC、及びMUXノードを通じて等価性リストを伝搬させる技術については後述する。 $SSGT$ のPOノードに到達すると、

(22)

42

アルゴリズムは、 $BSGT$ 内の対応するOUTノードで等価性が確定しているか、及び、対応する条件がトリグジャーであるかどうかをチェックする。そうでない場合、アルゴリズムは、 $SSGT$ と $BSGT$ は等価でないことを報告する。 $SSGT$ のすべてのOUTノードに対して無条件等価性が得られた場合に限り、アルゴリズムは、 $SSGT$ と $BSGT$ が等価であると宣言する。

【0105】等価関係は、OPノードを通じて以下のよう伝搬する。 $SSGT$ 内のOPノードと、同じ演算を実行する $BSGT$ 内のOPノード u で、 v の入力が u の対応する入力と条件付き等価関係を有するようなものが存在する。このような場合、 v と u の出力は、対応する入力の等価条件の論理積と等価である。LOGICノードに遭遇した場合、等価性リストをその出力に伝搬させるのではなく、 $BSGT$ 内の基底変数の関数としてその出力を表すようにBDDを構成する。これを行う理由は、LOGICノードはスケジュールにおいて変換または導入されることがあるため、 $SSGT$ と $BSGT$ の等価性を証明するためには解釈される必要があるからである。2人MUNIXノードの1 (0) データ入力からその出力へ等価性リストを伝搬させることは、選択 (セレクト) 信号に対するBDDを取得し、それ (その補数) と、データ入力の等価性リスト内のすべての条件との論理積をとることによって、行われる。

【0106】4.2 一般的な場合のスケジュール検証アルゴリズム このサブセクションでは、一般的な場合のアルゴリズムの好ましい実施例について説明する。このアルゴリズムのタスクは、スケジュール及びビヘイビアの出力間の無条件等価性を確定することである。 $SSGT$ が非巡回 (無閉路) であれば、セクション4.1のシンボリックシミュレーションに基づく等価性チェックで十分である。フィードバック (ループ) が存在する場合、等価性チェックアルゴリズムが有用であるためには、ループが完了するまで反復せずに2つの記述の等価性を検証することが必要である。ループを扱うため、アルゴリズムは、ループ不変項を抽出する。不変項は、ループ終了点におけるスケジュールとビヘイビアの間の変数対応である。不変項抽出は、等価性の証明を生成するためにループを完了まで反復することを必要にしない帰納法に基づく。すべてのループ不変項が抽出されない場合、等価性チェックはフォールスネガティブを返す可能性がある。等価性チェックアルゴリズムは、スケジューリングが前に定義した意味で代表的である場合、すべてのループ不変項を抽出し、真の否定及び肯定を返すことを保証する。このアルゴリズムは、既った肯定を返すことがないという意味で安全である。本発明の検証アルゴリズムについて説明するための例を提示し、その後でその詳細について説明する。

【0107】4.2.1 具体例

アルゴリズムは、入力として、ビヘイビア及びスケジュー

(23)

43

ール状遷移グラフ (STG) 表現 (それぞれBSTG 及び SSG) をとる。STGに加えて、プライマリ入力及び出力の対応のリストも、アルゴリズムに入力として提供される。アルゴリズムは、SSG内の小さいバスセットを列挙することによって動作する。これらのバスは、SSGとBSTGの間の等価性を証明するための基礎として使用される。

【0108】例3: 図5 (a) に、簡約 (reduced) SSG (傳遞構成成分を抽出し無閉路パスをつづしたもの) の例を示す。このSSGに対して、次の状態列を列挙することができる。

(AE, ABCE, ABCDCE, ABCDCDC
E, ...)

なお、バス [ABCDCE, ABCDCDC, ...] は、ループ本体の異なる回数の実行に対応する。これらのバスのすべてをBSTG上でシミュレートする必要がある。ノード [C] は、ループ終了点に対応する。

(C) が0、1、及び3回現れるバスの数を列挙する。これは、ループに全く遭遇しないこと、ループ終了後に遭遇するがループ本体には遭遇しないこと、及びループ本体を2回実行すること、にそれぞれ対応する。最初の2つは単純バスであり、明確に列挙すべきである。終了バスが3回現れるバスを列挙する理由は、ループ本体を2回実行することによってループ不変項の生成に対する問題を認定することである。従って、この例で列挙されるバスは [AE, ABCE, ABCDCDC E] である。

【0109】これらのバスのそれぞれについて、BSTGの対応するバスをシミュレーションにより取得する。次に、アルゴリズムは、SSG及びBSTGの対応するバスが等価であることを証明する。ループ本体を含むバス [ABCDCE] に対して、アルゴリズムはさらに読み、ループ本体内の演算が任意回数実行された場合に、SSG及びBSTGの対応するバスどうしの間の等価関係が依然として維持されるかどうかを帰納的に証明する。これを行うため、アルゴリズムは、カット点 [ABCDCE] 及び [ABCDCEDC] などの変数対応を抽出する。この場合、カット点 [ABCDCE] 及び [ABCDCEDC] における対応のバスは同一のままである。従って、帰納法により、列 [DC] を任意回数だけ [ABCDCE] に置き換えても依然として変数対応は維持されるということができ、従って、[ABCDCEDC] と、対応するBSTGバスが等価である場合、ループ本体の任意回の反復に対して、SSGとBSTGは依然として等価になる。

【0110】次に、ループ終了に対応するカット点での対応する変数のセットが同一のままにならないような、別のシナリオを考える。セットが変わると、異なる結果を避けるために、収束するまで反復する必要がある。

【0111】例4: 図6に示す例は、フォールスボジテ

44

ィブ (制った肯定) を避けるために収束するまで反復する必要がある理由を例示する。図6の (a) 及び (b) は、それぞれ、回路のビヘイビア及びスケジューラを示す。なお、これらの2つは、スケジューラの状態におおける文 $c = d + 2$ のため、対応しない。最初に識別されるループ本体を含むバスは [1, 2, 3, 4, 5, 2, 3, 4, 5, 2, 6] である。図6 (b) における状態2は、ループ終了として識別され、状態3、4及び5はループ本体として識別される。ループの1回の実行のシンボルシミュレーションの後、得られる変数対応は $\{a = p, b = q, d = s\}$ であり、2回の実行の後に $\{a = b\}$ である。なお、ループの後の状態6の文は $\{b = q\}$ をシミュレートするための対応セットとしてこれらのいずれれを用いても、bとqが等価であると見なされていることによりフォールスボジティブを引き起こすことになる。

【0112】3回の実行の後にはじめて、手続はbとqが対応しないと判定し、スケジューラとビヘイビアは非等価であると見なすことができる。このように、この場合、収束に到達するには、最初のバスのもう1回の反復をシミュレートしなければならない。すなわち、バス [1, 2, 3, 4, 5, 2, 3, 4, 5, 2, 3, 4, 5, 2, 6] もシミュレートする必要がある。

【0113】例5: 最後にもう1つのシナリオを考える必要がある。前の例で、ループ反復後の対応のセットは、収束に到達するまで単調に減少した。しかし、一般に、反復を多く実行するにつれてこのセットが仕様に倣化するような例が考えられる。なお、実質的な例は、ループ本体のすべての反復回数について等価性をチェックすることである。従って、収束が得られるまですべての極小対応セットを追跡しなければならない。これは、他の対応セットのスーパーセットでないセットに対応する。ループ本体に続くコードは、収束後に得られる対応セットに対してシミュレートされるのに加えて、すべてのこのような極小セットに対してシミュレートされる。なお、この追加の対応によりシミュレートしないことによってフォールスボジティブを生じる可能性がある。後述するように、これらの極小対応セットを用いたシンボルシミュレーションは、等価性をチェックするために必要十分である。

【0114】4. 2. 2. アルゴリズムの問題

図7は、一般的な場合を扱う本発明の方法の好ましい実施例の疑似コードを示す。このアルゴリズムの第1のタスクは、SSGの、ループを構成する部分を識別する必要がある。ループ不変項をループ終了点で計算する必要があるからである。ループは、強連結成分 (SCC: strongly connected component) を識別することによって見出される。各SCCは、1個以上の終了ノードを有し、そこからSCCの外へ遷移することが可能である。

その後、SCCを通らないサブパスを併合して、以後列

(24)

45

挙する必要があるノード及びバスの総数を減らすためにSSSGをつづす。図5 (a) に、これらのステップによってSSSGがどのように影響されるかが示されている。これらのステップの結果、状態Cは、状態C及びDからなるSCCの終了点として識別される。

【0115】図7における疑似コードの第4行は、簡約SSG内のバスを列挙するループの開始をマークする。このバスは、BSTG内の対応するバスに対してチェックされなければならない。バス列挙前にSSSGを簡約することにより、大幅にバスは少なくなる。図5 (a) のSSSGの場合、最初に列挙される3個のバスは、次の状態列からなる。

(AE, ABCE, ABCDCDC E)

これらのバスのそれぞれについて、BSTG内の対応するバスをシミュレーションにより取得する。図7の疑似コードの第5行は、列挙されたSSSGバスに対するRTL回路 (SSGという) を取得する。第5行は、また、SSSGバス内の終了ノードへの各遷移に対応するRTL回路内のバスを識別する。カットとは、本明細書においては、状態遷移によりある状態から別の状態へ伝搬する変数のセットとして定義される。図7の疑似コードの第6行は、SSSGバスを列挙する

に必要となるすべての状態遷移判定をカプセル化するバスシグナル (Pathsignal) という信号を生成する。SSSGバスに対応するBSTG内のバスを識別するシンボルシミュレーションは、図7の疑似コードの第7行で、手続をConstraintedsymbolicSimulation0によって、SSSGバスのPathsignalを用いて実行される。

【0116】図8を参照しながら、ConstraintedsymbolicSimulation0の詳細について説明する。BSTG内のループ状態から始めて、そのタスクは、Pathsignalと両立する遷移により到達可能な状態を識別することである。到達した各状態で、対応する信号を識別するためには非軽減シンボルシミュレーションを実行する (図8の第5行)。次に、その状態からの、Pathsignalと両立する出遷移を識別する。このプロセスは、BSTG内のEND状態に到達するまで続く。

【0117】図7の全体アルゴリズムに戻って、次のステップは、列挙されたバス内から不変項を抽出することである (図7の第8〜12行)。このステップは、ループが存在しないときに不要となる。バスに沿って遭遇する各SCCに対して、図9に記載した手続をreturnloopinvariants0と呼び出す。この手続は、変数対応セットをcorrespsetlistとして返す。このリストのうち、前のシンボルシミュレーションの結果としてSCCカットで得られた変数対応より小さいを対応セットに対して、図7の第12行に示すように、SCCに続くバス部分のシンボルシミュレーションを再実行しなければならない。図7の第13行及び第14行は、得られた出力等価性が、バス条件以外の条件付きで

46

あるかどうかをテストする。そのように条件付きである場合、SSSGは等価でないと思える。出力が、列挙されたすべてのバスに対して無条件に等価である場合、SSSGは等価であると見なされる。

【0118】図9を参照すると、returnloopinvariantis0への入力は、列挙されたバスにおいてSCCの終了ノードに遭遇する3つのインスタンスに対応するSSG内の3個のカット (ssgcuts 1, 2, and 3) である。この手続は、ループに続くバス部分のシンボルシミュレーションが実行されなければならないような対応セットのリストを返す。図9の手続の第1行は、ssgcutに対応するBSTGにおける変数 (bagcuts) というを取得する。第2行及び第3行は、BSGにおいて抽出されたカットどうし間の2つの回路を抽出し、カット2とカット3の間の回路が、カット1とカット2の間の回路の直なる別のインスタンス (コピー) であるかどうかを確かめる。そうでない場合、対応がないと思える。適当なcorrespsetlistが返される。カット間の回路が同型である場合、非自明な対応セットが存在する可能性がある。

【0119】このセットを見つけたため、手続は、最初にカット2から始めて、一度に1回のループ実行 (すなわち、2つのカットの間の部分) だけ進むように、SSGとBSGをシンボルシミュレーションする。各シミュレーションの後には得られる変数対応 (correspset_{next}) を、そのシミュレーションの最初における対応 (correspset₀) と比較する。これらのセットが同一である場合、これは要求された固定点であり、手続は、この点で見出した対応した変数対応 (correspsetlist) を一部として返す。そうでない場合、correspsetlistを初期変数対応として、1ループ実行のシンボルシミュレーションを繰り返す。この手続はまた、1回の実行のシンボルシミュレーションが生成されたときには、この対応セットをcorrespsetlistに追加する。これは、例4及び例5において議論したようなフォールスボジティブを避けるためである。

【0120】すべての変数対応を識別するのに要する反復回数は、可能な変数対応の総数によって制限される。最悪の場合、これは、SSGとBSG内のループ本体の変数の個数の積になりうる。実際には、変数対応の数は、変数の個数に關して線形であり、ほとんどの対応は、最初の実行自体の後に見出される。従って、この手続は、有限回の (実際には、非常に少ない) ループ反復でループ不変項を得る手段である。

【0121】4. 2. 3. アルゴリズムの正当性及び有効性

フォールスボジティブは、2つの表現が実際には等価でないときに、検証ツールがそれらを等価であるとみなす場合に生じる。フォールスボジティブは、2つの表現が

(25)

47

・実際には等価であるときに、検証ツールがそれらを等価でないといみなすときに生じる。次の定題は、本発明のアルゴリズムを特徴づける。

【0122】定題2：図7の手続きCompareSTGsは、

(a) 代表的スケジューリング、及び、(b) 実現不可能な反復カウンタによるネガティブの可能性がない、という仮定の下で、フォールスポジティブまたはフォールスネガティブを発生しないことが保証される。

【0123】(証明) スケジュール及びビヘイビアがいずれも非巡回的であるとき、フォールスポジティブが発生しないことは、基本的なシンボリックシミュレーションに基づく等価性チェッカの性質である。フォールスネガティブは、非巡回的である場合、シンボリックシミュレータによって非解釈とされる同様の機能の知識が最適化で利用されるときにのみ発生しうる。残りの解析では、シンボリックシミュレーションに基づく等価性チェッカは、非巡回パスにおいて正しい変数対応を見出すという事実に依拠することができる。

【0124】さらに興味深いことは、ループが存在する場合に本発明のアルゴリズムでいつフォールスネガティブ及びポジティブが発生し得るかの解析である。ビヘイビア記述は非巡回的(ループを含む)であるがスケジューリング記述は非巡回的である場合、代表的スケジューリングは必ずしも、両方の記述にループがある場合、生成で許容されない。両方の記述は、フォールスポジティブされる変数対応が多すぎるときにフォールスポジティブが起こり、生成される変数対応が少なすぎるときにフォールスネガティブが起こる。

【0125】まずフォールスネガティブを考える。ループ停止の正当性は、本発明の手続きでは、スケジューリング及びビヘイビアにおけるループの停止条件どししの間の対応を決定することによってチェックされる。本発明のアプローチは、実現不可能な反復カウンタについて知らない、スケジューリング記述を生成するために使用される最適化が実現不可能な反復カウンタの知識を使用する場合、本発明の手続きはフォールスネガティブを報告する可能性がある。また、実現不可能な反復の反復の後にミループ本体内の間の光が「居性化」されるときにこれは起こり得る。従って、代表的スケジューリングであり、かつ、実現可能な反復カウンタがないという仮定の下では、このようなフォールスネガティブは起こり得ない(証明略)。

【0126】図9に示すように、不変項抽出手続きは、反復のたびにに変化しない変数対応のセットを識別するまで、ループを収束するまで反復する。このプロセスで生成される各極小変数対応セット(これは、他の対応セットのサブセットではない)は、ループ終了点の後のコードのシミュレーションを実行するために明々に使用される。ループのn回の実行後に得られる変数対応セットをCS_nで表す。以後のシミュレーションで使用される、

48

極小変数対応セットの集合を{CS_i}で表す。すなわち、CS₀ ⊆ {CS_i}は、極小対応セットであり、ループ終了以後のシンボリックシミュレーションのために使用される。

【0127】明らかに、CS₀ ⊆ {CS_i}でのシミュレーションにより生じるネガティブの等価性結果は、実現不可能な反復カウンタがないという仮定と、シンボリックシミュレーション手続きの基本的性質により、真のネガティブである。

【0128】次に、フォールスポジティブを考える。変数対応のセットの固定点(すなわち、CS₀ = CS_{fix})に到達するのにk+1回の反復が必要であると仮定する。帰納法により、このセットは、n ≥ kに対して、n回の反復に対応する実行されたパスに対する正しい変数対応のセットであるということが出来る。従って、CS₀でのシミュレーションの後に得られるポジティブの等価性結果は、n ≥ kのすべてのnに対して真のポジティブである。

【0129】ここで、スケジューリングにおけるkより少ない同数のループの実行に対応するパスを考える。すなわち、n < kとする。アルゴリズムは、n回の反復後の終了をチェックするように正しく動作すること、すなわち、この場合にフォールスポジティブがないことを示す必要がある。考慮すべき次の2つの場合がある。

【0130】1. CS₀は、極小対応セットのうちの1つである。すなわち、CS₀ ⊆ CS_iである。この場合、すべての極小対応セットは、ループ終了点以後明示的にシミュレートされるため、CS₀はフォールスポジティブを発生し得ない。

【0131】2. CS₀は、極小対応セットのうちの1つでない。極小対応セットの定義により、CS₀は、{CS_i}内の対応セットのうちの1つのスーパーセットでなければならない。この場合、{CS_i}内のすべての対応セットでのシンボリックシミュレーションがポジティブの結果を発生する場合、CS₀でのシンボリックシミュレーションも同様となり、CS₀や別個のシミュレーションをする必要はない。(他方、{CS_i}内のいづれかの対応セットでのシンボリックシミュレーションがネガティブを発生する場合、記述は非等価であり、CS₀でのシンボリックシミュレーションは意味がない。)

【0132】[4. 2. 3. 1. ネストしたループの展開] ネストしたループを扱うためには、内側のループに入るたびに不変項を解析しなければならない。前述のCompareSTGs手続きに加えて、ループネスティングを決定する解析が必要となる。ネストしたループをどのように扱うかについての直観的な説明は、セクション5. 2のネストしたループの例のケーススタディを参照。

【0133】[4. 3 アルゴリズムの効率] アルゴリズムの効率は、基本的に、次の3つのファクタから構成

49

出される。

(1) データパス状態は列挙されない。
(2) 断片は断片されない。
(3) 不変項を抽出するためにループは完了まで反復されない。

【0134】ファクタ(1)及び(2)は、等価性チェッカのためのアルゴリズムの内側ループで使用する非解釈シンボリックシミュレーションに含まれる。ファクタ(3)は、本発明の不変項抽出アルゴリズムによる。手続CompareSTGs(0)におけるSSTG内のSCC識別及び終了点の識別は、SSTGのサイズに関して線形である。CompareSTGs(0)におけるパス列挙は、つぎのSSTGに対して行われる。これは、最悪の場合に列挙されるパスの個数が、SSTG内の状態数ではなくスケジューリングの個数に比例して指数関数的になることを意味する。このパス数は一般に非常に小さい可能性が高い。不変項を抽出するために、列挙に要するループ反復回数は、可能な変数対応関係の個数によって制限される。最悪の場合、これは、ループ本体内の変数の個数に比例して2次に成り得る。実際には、スケジューリングにおけるエラーがないとき、代表的スケジューリングの場合、すべての変数対応は、2回のループ反復の列挙により見出される。シンボリックシミュレーションのブール演算には二分決定ダイナミクス(BDD)が必要とされるが、このような2倍反復は実際には非常に小さいため、BDD生成がボトルネックとなることはない。シンボリックモデルチェッカ(symbolic model checking)のような技術に比べて、本発明のアルゴリズムのランタイム計算量(複雑さ)は小さく、本発明のアルゴリズムは、取り組んでいる特定の検証問題に対する高速なカスタマイズされた解決法として適している。

【0135】[4. 4 スケジューリング検証システム] 本発明の重要な特徴は、同様のスケジューリング記述から得られるような同様のスケジューリングの正当性をチェックするシステムとして実現される。この正当性をチェックするシステムとして実現される。ループ不変項抽出器14. 1は、ループが存在するときに非巡回スレッドの十分なセットを決定する。シンボリックシミュレータ14. 2は、ループ不変項を抽出する。等価性証明器14. 3は、非巡回スレッドの等価性を証明する。このシステムは、

- ・ サイクル消費の増大
- ・ 消費並べ替え
- ・ ループの展開、巻付け、折畳み及びパイプライン化
- ・ 演算の段階実行

のうちの1つ以上により変換されたビヘイビア記述を扱うことが可能である。

【0136】本発明のもう1つの重要な特徴は、同様のビヘイビア記述に対して同様のスケジューリングを検証するシステムとして実現される。このようなシステムの好ま

(26)

50

しい実施例を図16に示す。スケジューリング状態遷移グラフ生成器15. 2は、15. 1からスケジューリングを受け取り、スケジューリング状態遷移グラフとして指定する。ビヘイビア状態遷移グラフ生成器15. 3は、同様のビヘイビアをビヘイビア状態遷移グラフとして指定する。スケジューリングスレッド15. 4は、スケジューリングを受け取り、スケジューリング状態遷移グラフから対応するビヘイビアスレッドを選択する。変換器15. 6は、スケジューリングスレッドをスケジューリング構造グラフに、及び、ビヘイビアスレッドをビヘイビア構造グラフに変換する。等価性チェッカ15. 7は、前記スケジューリング構造グラフと前記ビヘイビア構造グラフの等価性をチェックする。

【0137】[4. 5 スケジューリング検証コンビュータシステム] コンビュータは、本発明の技術を実現するための非常に有効な手段である。本発明の技術を実現するこのようなコンビュータシステムはまた本発明の技術的範囲に入る。このようなコンビュータは、プロセス及びメモリを有する。メモリは、コンビュータが同様のスケジューリングの正当性をチェックすることを可能にする命令を含む。ここで、同様のスケジューリングのビヘイビア記述から得られる。具体的には、メモリ内の命令は、ループが存在するときに非巡回スレッドの十分なセットを決定するためにループ不変項を抽出する命令を含む。さらに、命令は、ループ不変項を抽出するためのシンボリックシミュレーションの命令を含む。さらに、命令は、非巡回スレッドの等価性を証明する命令を含む。

【0138】なお、コンビュータは、PC、メインフレーム、ワークステーションあるいはネットワーク上のリモートコンビュータを含むいかなる種類のコンピュータとすることも可能である。

【0139】コンビュータシステムの好ましい実施例は、命令を含むメモリを有するコンピュータからなる。この命令は、コンビュータが、図4に示した疑似コードを実行することを可能にする。別の好ましい実施例は、コンビュータが、図7～図9に示した疑似コードを、単独に、またはすべての可能な組合せで、実行することを可能にする命令を含むメモリを有するコンピュータからなる。

【0140】なお、命令は、高水準言語、低水準言語、アセンブリ言語及び機械語を含む(これらに限定されない)任意の形式とすることが可能である。

【0141】[4. 6 スケジューリング検証コンビュータプログラム製品] 本発明の重要な特徴は、コンビュータプログラム製品として実現される。このプログラム製品は、コンビュータが同様のスケジューリングの正当性をチェックすることを可能にする命令を有するコンピ

(27)

51

ユーティリティ媒体を含む。なお、コンピュータ可読媒体は、フロッピー（登録商標）ディスク、ハードディスク、CD、チップ、テープ、IC付きカードリッジなどを含む（これらに限定されない）任意の固定媒体を含む。コンピュータ可読媒体は、ネットワークを通じて伝送される、あるいは、インターネットからダウンロードされる命令も含む。

【101421】好ましい実装例では、コンピュータコードは、コンピュータが同様のスケジューリングの正当性をチェックすることを可能にする。ここで、回路に対するスケジューリングは、ベヘイブ記述から得られる。コンピュータコードは、ループが存在するときに非巡回スレッドの1/4分セットを決定するためにループ不変項を抽出するコンピュータコードと、ループ不変項を抽出するためのシンボリックリソニックコミュニケーションのコンピュータコードと、非巡回スレッドの等価性を証明するコンピュータコードを含む。

【0143】 コンピュータプログラム製品の好ましい実施例は、コンピュータコードを含むコンピュータ可読媒体を含む。このコンピュータコードは、コンピュータが、図4に示した制御コードを実行することを可能にする。別の好ましい実施例は、コンピュータが、図7～図9に示した疑似コードを、単独に、またはすべての可能な組合せで、実行することを可能にするコードを含むコンピュータ可読媒体を含むコンピュータプログラム製品からなる。

【0144】なお、コンピュータコードは、高水準言語、低水準言語、アセンブリ言語及び機械語を含む（これらに限定されない）任意の形式とすることが可能である。

101451

【発明の効果】 4. 結果：ケーススタディ1) 本発明を適用した結果について、本発明のアルゴリズムを我々の実験のスケジューリングの例に適用した詳細なケーススタディの形で提示する。これらの設計は、状態変数の個数及び算術演算の計算量（複雑さ）に関して十分に大々的であり、状態マージ等価性やシンボリックモデルチェックに基づいて従来のBDDによる検証アプローチでは確実に失敗する。

【0146】4. 7. 1. 状態スケジューリングの例
図10(a)に示すパイプラインSTGを考える。このパイプラインを、状態移行及びループ変換を含む最新のスケジューリングによってスケジューリングした。結果として得られたスケジュールSTGを図10(b)に示す。パイプライン内のループを。定常状態では(すなわち、ループが多数回実行されると仮定すると)、スケジューリングは、状態S6からなるSSCCをたどることに注意する。スケジューリングの動作を実行することがよくなる。定常状態では、ループの新たな反復がクロックサイクルに開始され、大きなフィードバック制御につながらず、

ことを示すことが可能である。

【0147】検証の観点から、この例で興味深い点は、この例は、セクション1、2で言及したほとんどの最適化（サイクル境界の導入、演算の並べ替え、パスセグメントの複製、ループペーパーライニング化、及び枚数実行）を同時に含むことである。これらの最適化は、スケジューリングの複雑さを大幅に増大させる。ベヘイアSTGのVHDL記述は、122行のコードからなり、7個の演算及び8個の中間変数（ブライマリ入力及びブライマリ出力力を除く）を含む。これに対して、スケジューリングSTGのVHDL記述は、289行のコードからなり、7個の演算及び54個の中間変数を含む。明らかに、ベヘイアのスケジューリングの構造的特徴性では、これらの等価な

性を証明することはできない。(データバス+制御) 状態空間をたどる VSI のような従来の低レベル (例: F5M 等) の状態空間では、データバス+制御の F5M 等値性チェックツールは非常に困難になる。ビヘイビア STG は 250 個の状態ビットを含む。スケジューラ STG はさらに大幅に多くの状態ビットを含む。R. K. Brayton et al., "VIS: A system for verification and synthesis", in Proc. In Conf. Computer-Aided Verification, July 1996, を参照。

【0148】図10(a)及び(b)に示す2つのSSTGの等価性を証明するために本発明の検証手続きで実行されるさまざまなステップについて説明する。図7に示した手続きComparativeTusの第4行により、SSTGで最初に列举されるスレッドは、(SA, SG)、(SA, SB, SC, SD, SE, SF, SG)、(SA, S, B, SC, SD, SE, SF, SG)及び(SA, SB, SC, SD, SE, SF, SG)及び(SA, SB, SC, SD, SE, SF, SF, S)である。

G)である。これらのスレッドのそれぞれについて、手続は、ビヘイビア及びスケジュールに対する構造グラミングを生成し、セクション3で説明したシンボリックシミュレーション手続きを用いてその出力の単独性を証明する。さらに、スレッド (SA, SB, SC, SD, SE, SF, SF, SG) については、ループ不要項を抽出する手続きを呼び出す。

【0149】 シンボリックシミュレーションがどのような
 方式でシミュレーションを実行するかを調べるために、
 ここでは、スレッドT1 = (SA, SB, SC, SD, SE, SF, SG) を考える。スケジューラ内構造
 図表 1 に示す。スケジューラ内の状態
 図表 2 に示す。スケジューラ内の状態
 図表 3 に示す。スケジューラ内の状態

【0150】手続きConstrainedSymbolicSimulationは、ビヘイビア、及び対応する構造グラフにおける、対応するスレッドを自動的に抽出する。結果として得られるビヘイビアスレッドは $T2 = (S0, S1, \dots, S7, S1, \dots, S7, S1, S8)$ であり、その構造グラフ (はSGT) は図12に示される。なお、スケジュー

(28)

5.3

ユールとトビエビの STG における状態境界は対応していないため、SSG_{IT} 内のカットに対応する BSG_{IT} の内には、等価関係をを用いて決定される。例えば、SSG_{IT} 内の第 1 のカットにおける値号に関する等価関係は、(t4, t4, 1), (W2, W2, 1), (c, c, 1), (i#1, i#1, 1), (i', i', 1), c) である。ここで、エンタリ $(s, 1, s, 2, cond)$ は、SSG_{IT} 内の値号 s 1 と BSG_{IT} 内の値号 s 2 が条件 $cond$ の下で等価であることを意味する。型別するように、値号 i と $i\#1$ は、BSG_{IT} 内の対応するカットを形成する。SSG_{IT} の第 5 及び第 6 のカットに対して BSG_{IT} 内で形成されるカットは、図 12 で点線を用いて示されている。これらのカットは、状態 SSG_{IT} を含むスケジュール STG ループの第 1 回の実行の開始及び終了を表す。これらの 2 つのカットのメンバである値号に関する等価関係は、(i', i', 1), (i#1, i#1, 1), (i#2, i#2, 1), (t4, t4, c), (t3#1, t3#1, 1), c) 及び (c', c', 1), (W2', W2', 1), (t4', t4', 1), c, c#1), (i', i#2, c, c#1) である。

【0151】上記のことから明らかに示されるように、ループ境界変数の多々についての対応は存在しない(例えば、b3、a1、a2、a3、a4、a5、a6、a7)。さらに解析すれば、b3は理解されるように、SSGの大部分(図中シェルの大部分)はシミュレートされていない。スケジュール内のループをもう1回展開することにより、影部分の変数の値については新たな等価関数が見出される(すなわち、ループ不変量のセットが増大する)。この例では、ループ変数間の等価関係が取除くためには全部で6回ループを展開する必要があることを示すことができる。なお、ベヘビア内のループは、スケジュールを抽出する際にスラブアラリーによって(偶然の一環ではなく)フアクタ6でデバライニングされたことに注目する。興味深い。

[0152] 4. 7. 2. X. 2.5 通信プロトコル
この例は、X. 2.5 通信プロトコルの送信(send)プロセスである。S. Bhattacharya, S. Dey, and F. Brglez, "Performance analysis and optimization of schedulers for conditional and loop-intensive specification", in Proc. Design Automation Conf., pp. 491-496, June 1994. を参照。演算への状態の直線的な割当てによるビヘイビアに対する時間フローグラフを図 13

(a) に示す。なお、これは配列変数を使用している。配列アクセスは、非解読可能であることが保証される。配列インデックスと配列名は対応することが保証される。これから得られる正しいスケジュールを図 1 3 (b) に示す。各状態内の数字は、その状態内で実行される演算に対応する。この例は 2 つの理由により興味深い。第 1 の理由は、状態 S 1 1 及び S 1 2 に対応するネストしたループである。第 2 の理由は、スケジューリングのパスに対して生成されるデータフローグラフは、ビヘイビアにおいて生成される対応するパスと構造的に同型ではないことである。図 1 3 (c) に、同じビヘイビアに対する正

54

をしくないスケジューリングを示す。ループのネスティングを見出すためには、スケジュールのSTGに対する正規表現(regular expression)を導出する。Z. Kohavi, "Switching and Finite Automata Theory", McGraw Hill Book Company, second ed., 1978. を参照。

【0153】まず図13(b)を考える。これに対して
導出される正規表現は $S_0S_1S_2S_4(S_3S_4)^*(S_5S_4)^*$
 $(S_3S_4)^*$ * S_6 である。上付きの s のある各部
分表現はループ本体を構成する。これは明順にループに遭遇
するたびに、そのループに対してする不変項を識別しなけれ
ばならない。例えば、部分表現 $S_0S_1S_2S_4(S_3S_4)^*$
 $(S_3S_4)^*$ * において、内側ループ $(S_3S_4)^*$ の不変項
は、外側ループの不変項と区別するために外側ループに
要求される反復回数と同じ回数だけ抽出されなければなら
ない。手続きの残りの部分は端と同じであり、第1の
ケーススタディの通りに従う。

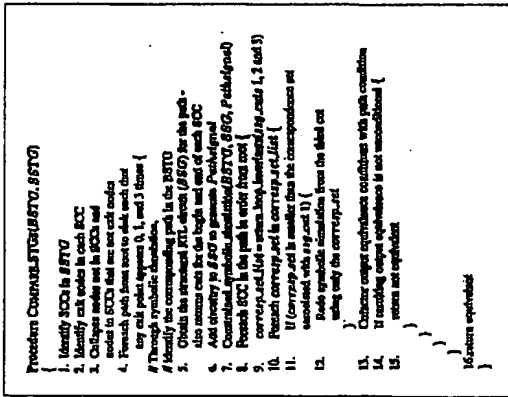
【0154】次に、図13 (c) を考える。導出される正規表現は、 $S_0S_1S_2S_4^* (S_3S_2S_4^*)^* + (S_0S_2S_4^*)^* S_6^*$ である。この正規表現は、外側ループの内側ループに共通の部分表現を有する2個のループを含めという点で前のものは異なる。部分表現 $(S_3S_2S_4^*)^*$ における、外側ループのループ不変項を決定することは興味深い。これを達成するたぐ、要素対応が安定するまで、外側ループの有数の反復が詳細に、内側ループのインスタンスタンス化のすべての可能な組合せが考慮される。例えば、外側ループが部分表現 $(A^* + B^*)^*$ である場合、外側ループの2回の実行は、4個のパス、すなわち、

$A^*A^*, A^*B^*, B^*A^*, B^*B^*$
 に沿って変数対応を計算しなければならぬことを意味する。なお、各パスは、それぞれネスティングのないループからなる。外側ループの3回目の実行で、列挙されるパスは次のようになる。

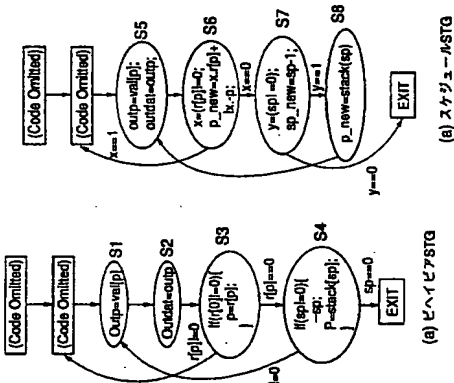
外側ループに対するループ不変性を証明するためには、外側ループの n 回目の実行における部分パス P_n から生成される要約対象が、 P_n から導出される $n+1$ 回目の実行におけるすべての部分パスから生成される 1 対目の実行に対しては、外側ループの 2 回目の実行後に A^* から生成されるループ不変性は、3 回目の実行において A^*A^* 及び $A^*A^*B^*$ から生成される要約と同じと見ておけるスケジューリング (c) におけるスケジューリングは、状態 S 3 から出る遷移の正しいくない実現の結果、要約には正しくない。これは、本発明の手続きでは検出される。2 つの表現における da は要約の間に対応を成立させることが不可能であるからである。

(33)

[図 7]

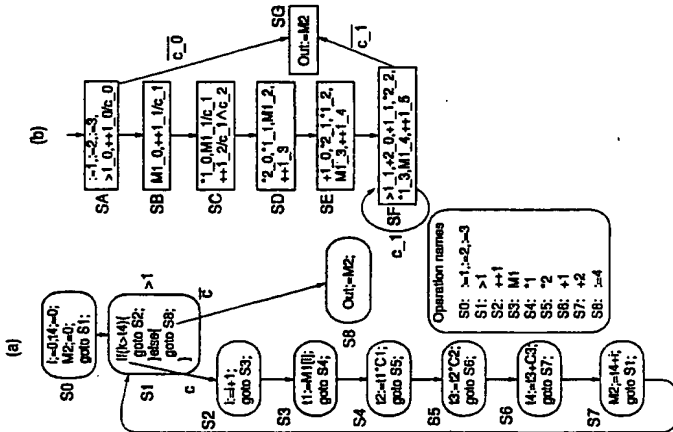


[図 1 4]



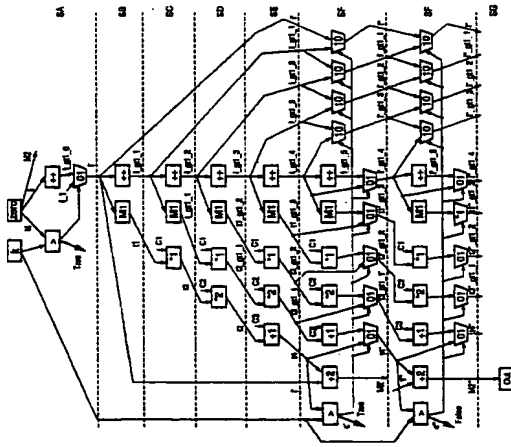
(34)

[図 1 0]



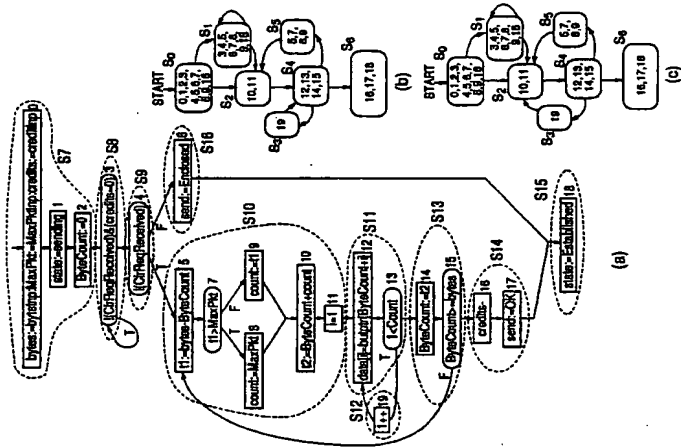
(35)

【図11】

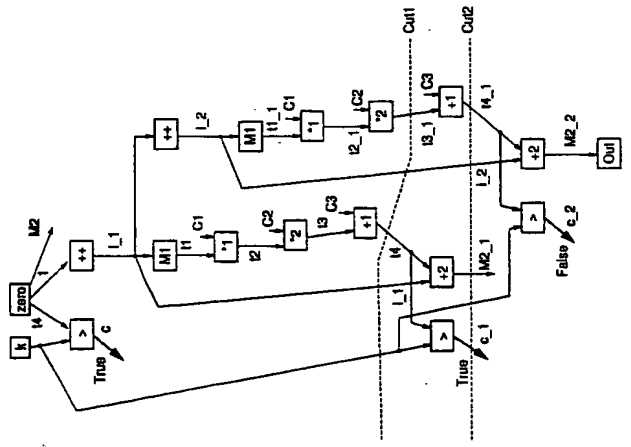


(36)

【図13】

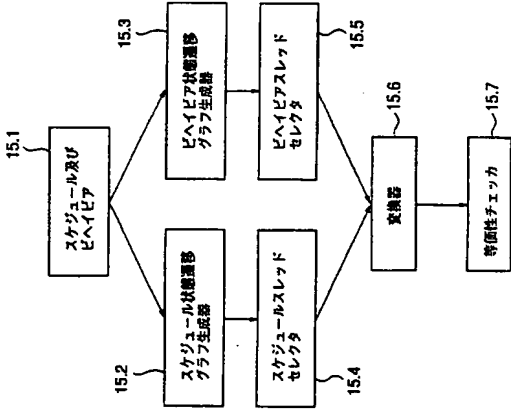


【図12】



(37)

【図16】



フロントページの続き

(72)発明者	スブラジット・バクチャリヤ	(72)発明者	アナンド・ラグナサン
	アメリカ合衆国, ニュージャージー		アメリカ合衆国, ニュージャージー
	08540 プリンストン, 4 インディペン		08540 プリンストン, 4 インディペン
	デンス ウエイ, エヌ・イー・シー・ユ		デンス ウエイ, エヌ・イー・シー・ユ
	ー・エス・エー・インク内		ー・エス・エー・インク内
(72)発明者		(72)発明者	アーティ・グプタ
			アメリカ合衆国, ニュージャージー
			08540 プリンストン, 4 インディペン
			デンス ウエイ, エヌ・イー・シー・ユ
			ー・エス・エー・インク内
Fターム(参考)	5B046 BA08 BA03 JA01 JA04		